

Thomas Wenzlaff

Microsoft JScript

für den

Hobby-Programmierer

Stand 20.03.2007



Hinweise zum Dokument

Wer strukturiert und objektorientiert programmieren möchte, sollte sich z.B. mit Javascript/JScript beschäftigen.

Schwerpunkte dieser Dokumentation sind:

Programmierung mit Javascript/JScript unter dem Betriebssystem "Microsoft Windows 32-Bit"
und mit JScript verbundene **programmtechnische Verwendungen** der Produkte

"Microsoft Internet Explorer"	unter und ab der Version 6.x. (vor allem ab 6.x),
"Microsoft Windows Media Player"	exemplarisch Version 7.1
"Netscape"-Browser	unter und ab der Version 6.x.

Dieses vorliegende Dokument dient nicht als Lehrmaterial, sondern ist eine stark strukturierte Systematik. Sämtliche Beispiele dienen ausschließlich dem Verständnis und als Anregungen zur Programmierung. Da sich das Dokument an den Hobby-Programmierer richtet, befindet sich in diesem Dokument nur eine eingeschränkte, aber ausreichende Syntaxbeschreibung.

Autor: Thomas Wenzlaff
www.twseite.de

Stand des Dokumentes: 20.03.2007

Rechtewahrung, Haftung und Verbesserungsvorschläge:

Das vorliegende Dokument richtet sich ausschließlich an den Nutzerkreis der Hobby-Programmierer.

Die Informationen in diesem Produkt werden ohne Rücksicht auf einen eventuellen Patentschutz verwendet. Warennamen, Hardware- und Softwarebezeichnungen, Softwarekomponenten sowie Algorithmen werden ohne Gewährleistung der freien Verwendbarkeit benutzt, gehören dem jeweiligen Eigentümer, oder sollten als solche betrachtet werden.

Der Autor kann für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen. Der missbräuchliche Einsatz dieser Dokumentation - vor allem bei sicherheitsrelevanter Programmierung - wird vom Autor grundsätzlich abgelehnt.

Für Verbesserungsvorschläge und Hinweise ist der Autor dankbar.

Warnung zur Zweckentfremdung von Javascript/JScript:

Es sei darauf hingewiesen, dass mit Javascript/JScript sicherheitsrelevante Aktionen programmierbar sind, die das Sicherheitsbedürfnis des Users betreffen und/oder rechtlich relevant werden können. Ein Problem kann die Unkenntnis des Benutzers sein, wenn o.g. Aktionen nicht für den User transparent gestaltet wurden.

Eine radikale Maßnahme gegen das Risiko ist z.B. die Abschaltung von Scripten (wie Javascript), ActiveX und Cookies (z.B. im Browser selbst oder anhand einer Firewall-Software). Die Abschaltung von Javascript durch den User macht die Anwendung dieser Dokumentation durch einen Programmierer hinfällig. Dem User sind daher dringend eine Antivirus- **und** eine Firewall-Software zu empfehlen, die **beide** auch während einer Netzwerksitzung im Internet aktiv sind. Z.B. erfolgen die Blockierung von nervenden Portscans durch Mitbenutzer jeder Art im Netzwerk und die Abwehr von immer häufiger eingesetzten Trojanern und Scriptviren (Scriptviren innerhalb eines HTML-Dokumentes oder einer HTML-Email werden durch die o.g. empfohlenen Softwares gefiltert).

Hinweise zur Kodierung von Quelltext in diesem Dokument:

Die im Dokument verwendete Rechtschreibung und Grammatik sowie das Layout des Dokumentes entsprechen nicht den aktuellen Rechtsnormen. In diesem Dokument werden auch aus Gründen der Transparenz DV-Fachbegriffe sowie Sprachstile verwendet, die nicht nur in der Sprachpflege umstritten sind. Der Ersatz deutscher Begriffe durch die aus der DV-Fachsprache ist nicht zu vermeiden. Konsequenz gesehen, hätte nur die englische Sprache verwendet werden dürfen, da die meisten Softwareentwicklungen und tendenziell neuen Produkte nicht aus dem deutschsprachigen Raum stammen.

Die in diesem Dokument verwendeten Einrückungen und Einschachtelungen von HTML-Elementen wie Tags, Scripten usw. dienen vorrangig der Transparenz z.B. von Quelltexten, Kommentaren und Syntax-Beschreibungen. Tippfehler lassen sich nicht vermeiden.

Diese Dokumentation wurde u.a. erstellt mit

Microsoft Office 97:

Leider hat sich die eingesetzte Version für solche Arten von Dokumentationen als wenig geeignet erwiesen. Mit der kurz vor der tausendsten Seite erfolgten Meldung, dass die Textdatei komplett defekt war und der Autor auf Rat der Meldung den gesamten Text-Bestand über die Zwischenablage (!) in ein neues, leeres Dokument eingefügt hatte, wurde dieses klar: Auch das neue Dokument blieb defekt - Totalverlust der Daten. Die Indexerstellung über 1600 Seiten ließ die Software abstürzen (inklusive dokumenteigener Format-Vorlage). Fußzeilen funktionierten danach nicht mehr dokumentweit. Die Pflege eines Wörterbuches für die Rechtschreibungsanalyse erwies sich als nicht machbar, da Text-Formate nicht ausklammerbar waren.

Microsoft Office XP:

Die Leistungsfähigkeit der Software (auf Intel P 4 mit 1 GB Rambus, Win XP Home SP2) war am Ende, als sie ein Inhaltsverzeichnis formatiert erstellen sollte. Nach mehreren Programmabstürzen (inkl. während der Wiederherstellung des Dokumentes) sind wenigstens die Seitennummern-Felder richtig erhalten geblieben. Die zerstörte Formatierung musste manuell vereinheitlicht werden. Die Indexerstellung verlief tadellos.



Inhaltsverzeichnis

1.	Dialekte und Versionen von Javascript	17
1.1.	Javascript-Versionen beim Netscape	30
1.2.	Javascript-Versionen von Microsoft	30
1.3.	Feststellung des Dialektes von Javascript	36
1.3.1.	Schritt 1: Erkennung des Browserherstellers	36
1.3.1.1.	Browsererkennung anhand des Browsernamen (navigator.appName)	36
1.3.1.2.	Browsererkennung anhand der Unterscheidung browserinterner Objekte	37
1.3.1.3.	Browsererkennung beim Internet Explorer ab IE 5.x	38
1.3.2.	Schritt 2: Erkennung der Javascript-Version (browserhersteller-spezifisch)	39
1.4.	Feststellung der aktuellen JScript-Maschine	40
2.	Javascript in HTML einbinden	40
2.1.	Javascript-Direktkodierung in das HTML-Dokument	40
2.1.1.	Javascript-Kodierung per HTML-Tag	40
2.1.2.	Javascript-Kodierung als Wert eines HTML-Attributes im HTML-Tag	41
2.1.3.	Javascript-Kodierung als Aktion eines Eventhandlers im HTML-Tag	41
2.1.4.	Javascript-Kodierung als Aktion eines Eventhandlers im SCRIPT-Tag	41
2.1.5.	Javascript-Kodierung zur Laufzeit des HTML-Dokumentes	42
2.1.5.1.	Methode eval()	43
2.1.5.2.	Methoden document.write() und document.writeln()	43
2.2.	Javascript-Kodierung in externer Datei *.js	45
2.3.	Javascript und Browserperformance	48
2.4.	Javascript- und HTML-Dateien auf dem Server	48
2.4.1.	robots.txt und Suchmaschinen	49
2.4.1.1.	robots.txt und ihre Lage auf dem Server	49
2.4.1.2.	robots.txt die nicht auf dem Server vorhanden ist (Scan-Standard)	49
2.4.1.3.	robots.txt und Aufbau	49
2.4.1.3.1.	robots.txt als Zeilenfolge-Script erstellen	49
2.4.1.3.2.	robots.txt und ihre Blöcke	49
2.4.1.3.2.1.	robots.txt-Block: Aufbau Zeile 1 (Zulassen bzw. Sperren von Suchmaschinen)	49
2.4.1.3.2.1.1.	robots.txt und Zeile 1: ALLE Suchmaschinen dürfen scannen	49
2.4.1.3.2.1.2.	robots.txt und Zeile 1: KEINE Suchmaschine darf scannen	49
2.4.1.3.2.1.3.	robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf scannen	49
2.4.1.3.2.1.4.	robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf NICHT scannen	49
2.4.1.3.2.2.	robots.txt-Block: Aufbau ab Zeile 2 (Zulassen bzw. Sperren von Inhalten auf dem Server)	49
2.4.1.3.2.2.1.	robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen gescannt werden	50
2.4.1.3.2.2.2.	robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen NICHT gescannt werden	50
2.4.1.3.2.2.3.	robots.txt und ab Zeile 2: Bestimmte Daten auf dem Server dürfen NICHT gescannt werden	50
2.4.1.3.2.2.3.1.	robots.txt und ab Zeile 2: Bestimmte Verzeichnisse auf dem Server dürfen NICHT gescannt werden	50
2.4.1.3.2.2.3.2.	robots.txt und ab Zeile 2: Bestimmte HTML-Dateien auf dem Server dürfen NICHT gescannt werden	50
2.4.1.3.2.2.4.	robots.txt und ab Zeile 2: NUR bestimmte Daten auf dem Server dürfen gescannt werden	50
2.4.1.3.3.	robots.txt und Beispiele	50
2.4.2.	.htaccess und .htpasswd und Passwortschutz	51
3.	Javascript-Elemente (Auswahl)	52
3.1.	Javascript-Bezeichner	52
3.2.	Javascript-Kommentar	53
3.3.	Datentypen	54
3.3.1.	array Datentyp (Basis-Datenstruktur)	54
3.3.2.	boolean Datentyp (Basis-Datentyp)	56
3.3.3.	date Datentyp (Basis-Datenstruktur)	56
3.3.4.	function Datentyp (Basis-Datenstruktur)	56
3.3.5.	Literal Datentyp (Basis-Datentyp)	56
3.3.6.	number Datentyp (Basis-Datentyp)	57
3.3.8.	Objekttyp oder Objektklasse (Basis-Datenstruktur)	58
3.3.9.	Zeigertyp (Basis-Datentyp)	58
3.4.1.	nicht Objektvariable (Variable nicht per new erzeugt)	66
3.4.2.	Objektvariable	67
3.4.2.1.	Objektvariable mit new deklarieren	72
3.4.2.2.	Objektvariable als Array Objekt aus Literalen deklarieren	74
3.5.	Operatoren	75
3.5.1.	Operatoren logische	75
3.5.2.	Operatoren arithmetische	75
3.5.3.	Operatoren bitweise (Bitoperatoren)	76
3.5.4.	Operatoren für Vergleich (Vergleichoperatoren)	76
3.5.5.	Operatoren für Verkettung von Zeichenketten	76
3.5.6.	Operator für Zuweisung	76
3.5.7.	Operator für Ermittlung des Datentyps (Operator typeof)	77
3.5.8.	Ausdruck berechnen, aber den Wert nicht liefern (Operator void)	77
3.5.9.	this (Zeiger auf aktuelle Objekt-Instanz)	77
3.5.10.	with (Zeiger auf aktuelle Objekt-Instanz)	77



3.5.11.	Operatoren in Microsoft JScript	77	
3.5.12.	Operatoren in Javascript 1.5 im Netscape 6.x	83	
3.6.	Anweisungen	84	
3.6.1.	Anweisungen in Javascript und JScript	85	
3.6.2.	Anweisungen nur in Microsoft JScript	99	
3.6.3.	Anweisungen in Javascript 1.5 im Netscape 6.x	101	
3.7.	Ausdruck (Expression)	102	
3.8.	Funktion	102	
3.8.1.	Funktion und optionale Argumente und Parameter	103	
3.8.2.	Funktion und optionaler Funktionswert	104	
3.8.3.	Funktion vordefiniert	106	
3.8.4.	Funktion durch Programmierer frei deklariert	106	
3.8.5.	Funktion als Objektkonstruktor (Objektklasse) per new	111	
3.8.6.	Funktion und Abarbeitungsfolge z.B. bei Rekursion	112	
3.8.7.	Funktion und Rekursionen	113	
3.8.7.1.	Übergabe von Variablen an die Rekursion	113	
3.8.7.2.	Rekursion und document.write() bzw. document.writeln() im HEAD	113	
3.1.9.	ASCII-Code und Unicode	113	
3.10.	Fehlerbehandlung in Javascript	114	
3.10.1.	Runtime Fehler (Laufzeitfehler) von JScript (Auswahl)	115	
3.10.2.	Syntax-Fehler von JScript (Auswahl)	115	
3.10.3.	Abfangen von Runtime Fehlern (Laufzeitfehlern)	128	
4.	Objekte in Javascript und im Browser	131	
4.1.	in Javascript vordefinierte Operationen mit Objekten	142	
4.1.1.	Operationen mit Objektinstanzen (Auswahl)	142	
4.1.1.1.	Ermittlung der Objektklasse einer Objektinstanz als Zeichenkette (typeof)	142	
4.1.1.2.	Vergleich von Objektinstanzen (valueOf) (Zeigervergleich)	143	
4.1.1.3.	Löschen einer Objektinstanz (incl. Speicherfreigabe) per null-Zuweisung	143	
4.1.2.	Standardmethoden aller Objekte in Javascript (Auswahl)	143	
4.1.2.1.	Boolean()	143	
4.1.2.2.	decodeURI() (IE ab 5.5, NS 6.x)	143	
4.1.2.3.	decodeURIComponent() (IE ab 5.5, NS 6.x)	143	
4.1.2.4.	encodeURI() (IE ab 5.5, NS 6.x)	143	
4.1.2.5.	encodeURIComponent () (IE ab 5.5, NS 6.x)	144	
4.1.2.6.	escape()	144	
4.1.2.7.	eval()	144	
4.1.2.8.	isFinite()	146	
4.1.2.9.	isNaN()	146	
4.1.2.10.	Number()	146	
4.1.2.11.	parseFloat()	147	
4.1.2.12.	parseInt()	147	
4.1.2.13.	String()	148	
4.1.2.14.	toString()	148	
4.1.2.15.	unescape()	149	
4.1.2.16.	valueOf()	149	
4.1.3.	Standard-Eigenschaften und -Methoden aller Objekte in Microsoft JScript	149	
4.2.	In Javascript vordefinierte Objekte (Script-Objekte)	160	
4.2.1.	arguments Script-Objekt	160	
4.2.2.	Array Script-Objekt	162	
4.2.2.1.	Array Script-Objekt mit Elemente eines beliebigen Datentyps	162	
4.2.2.1.1.	Array JScript-Objekt im Internet Explorer ab Version 5.5	170	
4.2.2.1.2.	Array Script-Objekt im Netscape ab Version 6.x (ab Javascript 1.5)	172	
4.2.2.2.	Array Script-Objekt aus Literalen	174	
4.2.3.	Boolean Script-Objekt	175	
4.2.4.	Date Script-Objekt	176	
4.2.4.	Enumerator JScript-Objekt des Internet Explorer	188	
4.2.5.	error JScript-Objekt im Internet Explorer	192	
4.2.6.	Function Script-Objekt	194	
4.2.7.	Math Script-Objekt	195	
4.2.8.	Number Script-Objekt	199	
4.2.9.	Object JScript-Objekt	208	
4.2.10.	String Script-Objekt	213	
4.3.	vordefinierte Objekte zum Browser (Auswahl)	220	
4.3.1.	Ansatz	220	
4.3.1.1.	vordefinierte Objekte in Javascript /JScript	220	
4.3.1.2.	Browserfenster und HTML-Dokument (Objekt window und Objekt document)	220	
4.3.1.3.	HTML-Dokument (Objekt document) und seine HTML-Elemente	221	
4.3.1.3.1.	HTML-Dokument und die Hierarchie der HTML-Elemente	221	
4.3.1.3.2.	HTML-Dokument und HTML-DOM	222	
4.3.1.3.3.	HTML-DOM	232	
4.3.1.3.3.1.	HTML-DOM beim Netscape 6.x (Übersicht)	232	



4.3.1.3.3.1.1.	Methoden vom Objekt document im HTML-DOM des Netscape	232	
4.3.1.3.3.1.1.1.	document.getElementById() des Netscape	232	
4.3.1.3.3.1.1.2.	document.getElementsByTagName() des Netscape	232	
4.3.1.3.3.1.1.3.	document.createElement() des Netscape	232	
4.3.1.3.3.1.1.4.	document.createAttribute() des Netscape	233	
4.3.1.3.3.1.1.5.	document.setAttribute() des Netscape	233	
4.3.1.3.3.1.1.6.	document.createTextNode() des Netscape	233	
4.3.1.3.3.1.1.7.	document.createTextRange() des Netscape	234	
4.3.1.3.3.1.2.	Objekt document.documentElement des Netscape	234	
4.3.1.3.3.1.2.1.	Eigenschaften von document.documentElement des Netscape	234	
4.3.1.3.3.1.2.2.	Methode von document.documentElement des Netscape	235	
4.3.1.3.3.2.	HTML-DOM beim Internet Explorer	238	
4.3.1.3.3.2.1.	thematisierte Zuordnung von Eigenschaften und Methoden des HTML-DOM zu	238	
4.3.1.3.3.2.2.	Eigenschaften zur Verwaltung des HTML-DOM im Internet Explorer	242	
4.3.1.3.3.2.3.	Methoden zur Verwaltung des HTML-DOM im Internet Explorer	249	
4.3.1.3.3.2.4.	Collectionen zur Verwaltung des HTML-DOM im Internet Explorer	266	
4.3.1.3.3.2.4.1.	attributes Collection des HTML-DOM im Internet Explorer	266	
4.3.1.3.3.2.4.2.	childNodes Collection des HTML-DOM im Internet Explorer	268	
4.3.1.3.3.2.4.3.	children Collection des HTML-DOM im Internet Explorer	271	
4.3.1.3.3.2.4.4.	tags Collection des HTML-DOM im Internet Explorer	275	
4.3.1.3.3.2.5.	command Objekt zum HTML-DOM des Internet Explorer	275	
4.3.1.3.3.2.6.	TextNode Objekt des HTML-DOM im Internet Explorer	280	
4.3.2.	window Objekt	281	
4.3.2.1.	window Objekt des Netscape (Übersicht)	283	
4.3.2.1.1.	Eigenschaften	284	
4.3.2.1.2.	Methoden	291	
4.3.2.1.3.	window.document Objekt des Netscape	304	
4.3.2.1.3.1.	Collectionen zum Objekt window.document des Netscape	305	
4.3.2.1.3.1.1.	window.document.anchors Collection des Netscape	306	
4.3.2.1.3.1.2.	window.document.applets Collection des Netscape	306	
4.3.2.1.3.1.3.	window.document.cookie Collection des Netscape (Cookie-Verwaltung im HTML-Dokument)	307	
4.3.2.1.3.1.4.	window.document.embeds Collection des Netscape	309	
4.3.2.1.3.1.5.	window.document.forms Collection des Netscape	309	
4.3.2.1.3.1.6.	window.document.frames Collection des Netscape	310	
4.3.2.1.3.1.7.	window.document.images Collection des Netscape	310	
4.3.2.1.3.1.8.	window.document.layers Collection des Netscape (nicht mehr ab Version 6.x)	310	
4.3.2.1.3.1.9.	window.document.links Collection des Netscape	311	
4.3.2.1.3.1.10.	window.document.plugins Collection des Netscape	311	
4.3.2.1.3.2	Objekte des HTML-Dokumentes des Netscape (Auswahl)	312	
4.3.2.1.3.2.1.	window.document.body Objekt des Netscape	312	
4.3.2.1.3.2.2.	window.document.frame Objekt des Netscape	312	
4.3.2.1.3.2.3.	window.document.frameset Objekt des Netscape	318	
4.3.2.1.3.2.4.	window.document.img Objekt des Netscape	324	
4.3.2.1.3.2.5.	window.document.layer / window.document.ilayer Objekt des Netscape unter 6.x	326	
4.3.2.1.3.2.6.	window.document.link Objekt des Netscape	332	
4.3.2.1.3.2.7.	window.document.span Objekt des Netscape	333	
4.3.2.1.3.2.8.	window.document.style Objekt und window.document.styleSheet Objekt des Netscape (CSS)	334	
4.3.2.1.3.2.8.1.	Style-Sheet-Deklarationen	337	
4.3.2.1.3.2.8.2.	Style-Sheet und vordefinierte Werte	340	
4.3.2.1.3.2.8.3.	Style-Sheet aus Datei einbinden	341	
4.3.2.1.3.2.8.4.	Style-Sheet und Wertzuweisung an Eigenschaften	343	
4.3.2.1.3.2.8.5.	Style-Sheet und Ausgabemedien (Pseudoklasse @media)	343	
4.3.2.1.3.2.8.6.	Style-Sheet und Seiten-Eigenschaften (Pseudoklasse @page)	343	
4.3.2.1.3.2.8.7.	Style-Sheet und HTML-Tag-bezogene Eigenschaften	343	
4.3.2.1.3.2.8.8.	Style-Sheet-Beispiele	353	
4.3.2.1.4.	window.event Objekt des Netscape	353	
4.3.2.1.4.1.	Eventarten (Auswahl)	355	
4.3.2.1.4.2.	Eigenschaften (Auswahl)	358	
4.3.2.1.4.3.	Methoden	359	
4.3.2.1.4.4.	Prinzipen der Eventbehandlung des Netscape	359	
4.3.2.1.4.4.1.	Event des Netscape einer Nicht-Standardbehandlung unterziehen	359	
4.3.2.1.4.4.1.1.	Event und Eventhandler dem Objekt window zuordnen (captureEvents(event_liste))	359	
4.3.2.1.4.4.1.2.	Event entlang der Eventhierarchie weiterreichen	360	
4.3.2.1.4.4.1.2.1.	Event entlang der Nicht-Standard- Eventhierarchie weiterreichen (handleEvent(event_objekt))	360	
4.3.2.1.4.4.1.2.2.	Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))	360	
4.3.2.1.4.4.2.	Event des Netscape von Nicht-Standardbehandlung wieder der Standardbehandlung unterziehen	360	
4.3.2.1.4.4.2.1.	Standard-Eventhandler dem Objekt window zuordnen (releaseEvents(event_liste))	360	
4.3.2.1.4.4.2.2.	Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))	360	
4.3.2.1.4.4.3.	Eventbehandlung durch eine Fremddatei mit signiertem Script	361	
4.3.2.1.4.4.3.1.	Beschaffung der Rechte "UniversalBrowserWrite" bzw. "UniversalBrowserWrite" per Privilegmanger	361	
4.3.2.1.4.4.3.2.	Eventbehandlung durch Fremddatei aktivieren (enableExternalCapture())	361	



4.3.2.1.4.4.3.3.	Eventbehandlung durch Fremde Seite deaktivieren (disableExternalCapture())	361
4.3.2.1.4.4.4.	Beispiel	361
4.3.2.1.4.5.	Beispiele zur Eventbehandlung des Netscape	362
4.3.2.1.4.5.1.	Formular	362
4.3.2.1.4.5.2.	Tastatur-Eventbehandlung beim Netscape	363
4.3.2.1.4.5.2.1.	Tastatur-Eventarten	363
4.3.2.1.4.5.2.2.	Tastatur-Eventeigenschaften	363
4.3.2.1.4.5.2.3.	Beispiel zur Tastatur-Eventbehandlung	364
4.3.2.1.4.5.3.	Mouse-Eventbehandlung beim Netscape	365
4.3.2.1.4.5.3.1.	Mouse-Eventarten	365
4.3.2.1.4.5.3.2.	Mouse-Event-Eigenschaften	365
4.3.2.1.4.5.4.	Lade-Ereignisse beim Netscape	365
4.3.2.1.4.5.4.1.	Lade-Ereignis für Bild	366
4.3.2.1.4.5.4.2.	Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)	366
4.3.2.1.5.	window.history Objekt des Netscape	366
4.3.2.1.6.	window.location Objekt des Netscape	366
4.3.2.1.7.	window.navigator Objekt des Netscape	367
4.3.2.1.7.1.	window.navigator.plugins Collection des Netscape	368
4.3.2.1.7.2.	window.navigator.mimeTypes Collection des Netscape	369
4.3.2.1.8.	window.screen Objekt des Netscape	370
4.3.2.2.	window Objekt des Internet Explorer	370
4.3.2.2.1.	window.clientInformation Objekt des Internet Explorer	410
4.3.2.2.2.	window.clipboardData Objekt des Internet Explorer	413
4.3.2.2.3.	window.dialogArguments Objekt des Internet Explorer	416
4.3.2.2.4.	window.document Objekt des Internet Explorer	418
4.3.2.2.4.1.	window.document.all Collection des Internet Explorer	447
4.3.2.2.4.2.	HTML-Elemente übergreifende Verwaltung im HTML-Dokument	448
4.3.2.2.4.2.1.	Verwaltung mehrerer HTML-Elemente gleicher Art im HTML-Dokument (Auswahl)	448
4.3.2.2.4.2.1.1.	window.document.anchors Collection des Internet Explorer (HTML-Element A (Anker))	448
4.3.2.2.4.2.1.2.	window.document.applets Collection des Internet Explorer	450
4.3.2.2.4.2.1.3.	window.document.body.timeAll Collection des Internet Explorer	451
4.3.2.2.4.2.1.4.	window.document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)	451
4.3.2.2.4.2.1.5.	window.document.form.elements Collection des Internet Explorer	452
4.3.2.2.4.2.1.6.	window.document.forms Collection des Internet Explorer	452
4.3.2.2.4.2.1.7.	window.document.frames Collection des Internet Explorer	453
4.3.2.2.4.2.1.8.	window.document.images Collection des Internet Explorer	454
4.3.2.2.4.2.1.9.	window.document.links Collection des Internet Explorer (HTML-Element mit HREF-Attribut)	455
4.3.2.2.4.2.1.10.	map.areas Collection des Internet Explorer	455
4.3.2.2.4.2.1.11.	window.document.select.options Collection des Internet Explorer	456
4.3.2.2.4.2.1.12.	window.document.selection.controlrange Collection des Internet Explorer	456
4.3.2.2.4.2.1.13.	window.document.selection.textrange Collection des Internet Explorer	457
4.3.2.2.4.2.1.14.	window.document.TextRange Objekt des Internet Explorer (Textbereich im Dokument)	458
4.3.2.2.4.2.1.14.1.	window.document.TextRange.TextRectangle Collection des Internet Explorer	460
4.3.2.2.4.2.1.14.2.	window.document.TextRange.TextRectangle Objekt des Internet Explorer	461
4.3.2.2.4.2.2.	Verwaltung mehrerer HTML-Elemente gleicher oder verschiedener Arten im HTML-Dokument	462
4.3.2.2.4.2.2.1.	allgemeine HTML-Element bezogene Verwaltung	462
4.3.2.2.4.2.2.1.1.	attribute Objekt des Internet Explorer (Attribute-Verwaltung für ein HTML-Element)	463
4.3.2.2.4.2.2.1.2.	attributes Collection des Internet Explorer	464
4.3.2.2.4.2.2.1.3.	childNodes Collection des Internet Explorer	467
4.3.2.2.4.2.2.1.4.	children Collection des Internet Explorer	471
4.3.2.2.4.2.2.1.5.	tags Collection des Internet Explorer	474
4.3.2.2.4.2.2.2.	spezielle HTML-Element bezogene Verwaltung	474
4.3.2.2.4.2.2.2.1.	Erzeugung ausgewählter eines HTML-Elemente durch Makro (command Objekt des Internet Explorer)	474
4.3.2.2.4.2.2.2.2.	Erzeugung eines privaten HTML-Elementes (custom Objekt des Internet Explorer)	479
4.3.2.2.4.2.2.3.	Cookie-Verwaltung im HTML-Dokument (document.cookie Collection des Internet Explorer)	484
4.3.2.2.4.2.2.3.1.	Zweck von Cookies	484
4.3.2.2.4.2.2.3.2.	Lage der Cookies am Beispiel von Microsoft Windows 9.x	486
4.3.2.2.4.2.2.3.3.	Vermeidung von Cookies	486
4.3.2.2.4.2.2.3.4.	Cookie verwalten (Beispiele)	486
4.3.2.2.4.2.2.3.5.	document.cookie Collection	488
4.3.2.2.4.2.2.4.	Zusätzliche Verhaltensweisen eines HTML-Elementes bzw. des HTML-Dokumentes	490
4.3.2.2.4.2.2.4.1.	Standard-Behavior	490
4.3.2.2.4.2.2.4.2.	element Objekt des Internet Explorer (HTC-Datei)	490
4.3.2.2.4.2.2.4.3.	behaviorUrns Collection des Internet Explorer	504
4.3.2.2.4.2.2.4.4.	document.namespace Objekt des Internet Explorer	504
4.3.2.2.4.2.2.4.5.	document.namespaces Collection des Internet Explorer	507
4.3.2.2.4.2.2.5.	Filter eines HTML-Elementes im Internet Explorer (filter Objekt des Internet Explorer)	508
4.3.2.2.4.2.2.5.1.	Einführung	508
4.3.2.2.4.2.2.5.2.	filters Collection des Internet Explorer	511
4.3.2.2.4.2.2.5.3.	Filtereigenschaften	511
4.3.2.2.4.2.2.5.4.	Filtermethoden	518



4.3.2.2.4.2.2.5.5.	Alpha Filter	518	
4.3.2.2.4.2.2.5.6.	AlphaImageLoader Filter	520	
4.3.2.2.4.2.2.5.7.	Barn Filter	520	
4.3.2.2.4.2.2.5.8.	BasicImage Filter	521	
4.3.2.2.4.2.2.5.9.	BlendTrans Filter	522	
4.3.2.2.4.2.2.5.10.	Blinds Filter	523	
4.3.2.2.4.2.2.5.11.	Blur Filter	523	
4.3.2.2.4.2.2.5.12.	CheckerBoard Filter	524	
4.3.2.2.4.2.2.5.13.	Chroma Filter	525	
4.3.2.2.4.2.2.5.14.	Compositor Filter	525	
4.3.2.2.4.2.2.5.15.	DropShadow Filter	526	
4.3.2.2.4.2.2.5.16.	Emboss Filter	526	
4.3.2.2.4.2.2.5.17.	Engrave Filter	526	
4.3.2.2.4.2.2.5.18.	Fade Filter	527	
4.3.2.2.4.2.2.5.19.	FlipH Filter	531	
4.3.2.2.4.2.2.5.20.	FlipV Filter	531	
4.3.2.2.4.2.2.5.21.	Glow Filter	531	
4.3.2.2.4.2.2.5.22.	Gradient Filter	531	
4.3.2.2.4.2.2.5.23.	GradientWipe Filter	532	
4.3.2.2.4.2.2.5.24.	Gray Filter	533	
4.3.2.2.4.2.2.5.25.	ICMFilter Filter	533	
4.3.2.2.4.2.2.5.26.	Inset Filter	533	
4.3.2.2.4.2.2.5.27.	Invert Filter	534	
4.3.2.2.4.2.2.5.28.	Iris Filter	534	
4.3.2.2.4.2.2.5.29.	Light Filter	535	
4.3.2.2.4.2.2.5.30.	MaskFilter Filter	535	
4.3.2.2.4.2.2.5.31.	Matrix Filter	536	
4.3.2.2.4.2.2.5.32.	MotionBlur Filter	537	
4.3.2.2.4.2.2.5.33.	Pixelate Filter	537	
4.3.2.2.4.2.2.5.34.	RadialWipe Filter	538	
4.3.2.2.4.2.2.5.35.	RandomBars Filter	539	
4.3.2.2.4.2.2.5.36.	RandomDissolve Filter	540	
4.3.2.2.4.2.2.5.37.	RevealTrans Filter	540	
4.3.2.2.4.2.2.5.38.	Shadow Filter	541	
4.3.2.2.4.2.2.5.39.	Slide Filter	542	
4.3.2.2.4.2.2.5.40.	Spiral Filter	543	
4.3.2.2.4.2.2.5.41.	Stretch Filter	544	
4.3.2.2.4.2.2.5.42.	Strips Filter	545	
4.3.2.2.4.2.2.5.43.	Wave Filter	546	
4.3.2.2.4.2.2.5.44.	Wheel Filter	546	
4.3.2.2.4.2.2.5.45.	Xray Filter	547	
4.3.2.2.4.2.2.5.46.	ZigZag Filter	547	
4.3.2.2.4.2.2.5.47.	Filter bei wichtigen Objekten (Auswahl)	548	
4.3.2.2.4.3.	HTML-Elemente im HTML-Dokument des Internet Explorer (Auswahl)	556	
4.3.2.2.4.3.1.	a Objekt des Internet Explorer	559	
4.3.2.2.4.3.2.	document.anchors Collection des Internet Explorer (HTML-Element A (Anker))	565	
4.3.2.2.4.3.3.	applet Objekt des Internet Explorer	566	
4.3.2.2.4.3.4.	document.applets Collection des Internet Explorer	570	
4.3.2.2.4.3.5.	area Objekt des Internet Explorer	571	
4.3.2.2.4.3.6.	bgsound Objekt des Internet Explorer	575	
4.3.2.2.4.3.7.	document.body Objekt des Internet Explorer	584	
4.3.2.2.4.3.7.1.	Eigenschaften beim Internet Explorer	589	
4.3.2.2.4.3.7.2.	Methoden beim Internet Explorer	591	
4.3.2.2.4.3.7.3.	document.body.timeAll Collection des Internet Explorer	593	
4.3.2.2.4.3.8.	button Objekt des Internet Explorer	593	
4.3.2.2.4.3.9.	comment Objekt des Internet Explorer	601	
4.3.2.2.4.3.10.	div Objekt des Internet Explorer	603	
4.3.2.2.4.3.11.	document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)	613	
4.3.2.2.4.3.12.	fieldSet Objekt des Internet Explorer	614	
4.3.2.2.4.3.13.	font Objekt des Internet Explorer	618	
4.3.2.2.4.3.14.	document.form Objekt des Internet Explorer	622	
4.3.2.2.4.3.14.1.	Objekt document.form.input und seine Varianten beim Internet Explorer	632	
4.3.2.2.4.3.14.1.1.	Objekt document.form.input.button des Internet Explorer	632	
4.3.2.2.4.3.14.1.2.	Objekt document.form.input.checkbox (Abhak-Kästchen) des Internet Explorer	633	
4.3.2.2.4.3.14.1.3.	Objekt document.form.input.fileupload des Internet Explorer	634	
4.3.2.2.4.3.14.1.4.	Objekt document.form.input.hidden des Internet Explorer	634	
4.3.2.2.4.3.14.1.5.	Objekt document.form.input.password des Internet Explorer	635	
4.3.2.2.4.3.14.1.6.	Objekt document.form.input.radio des Internet Explorer	635	
4.3.2.2.4.3.14.1.7.	Objekt document.form.input.reset des Internet Explorer	637	
4.3.2.2.4.3.14.1.8.	Objekt document.form.input.submit des Internet Explorer	638	
4.3.2.2.4.3.14.1.9.	Objekt document.form.input.text des Internet Explorer	638	



4.3.2.2.4.3.14.2.	document.form.elements Collection des Internet Explorer	641
4.3.2.2.4.3.15.	document.forms Collection des Internet Explorer	641
4.3.2.2.4.3.16.	frame Objekt	642
4.3.2.2.4.3.17.	document.frames Collection des Internet Explorer	651
4.3.2.2.4.3.18.	frameset Objekt	652
4.3.2.2.4.3.19.	document.html Objekt des Internet Explorer	662
4.3.2.2.4.3.20.	html comment Objekt des Internet Explorer	665
4.3.2.2.4.3.21.	iframe Objekt des Internet Explorer	665
4.3.2.2.4.3.22.	img Objekt	670
4.3.2.2.4.3.23.	document.images Collection des Internet Explorer	684
4.3.2.2.4.3.24.	input Objekt und seine Varianten	685
4.3.2.2.4.3.24.1.	input button Objekt	686
4.3.2.2.4.3.24.2.	input checkbox Objekt	690
4.3.2.2.4.3.24.3.	input file Objekt	695
4.3.2.2.4.3.24.4.	input hidden Objekt	699
4.3.2.2.4.3.24.5.	input image Objekt	702
4.3.2.2.4.3.24.6.	document.images Collection des Internet Explorer	707
4.3.2.2.4.3.24.7.	input password Objekt	707
4.3.2.2.4.3.24.8.	input radio Objekt	712
4.3.2.2.4.3.24.9.	input reset Objekt	716
4.3.2.2.4.3.24.10.	input submit Objekt	721
4.3.2.2.4.3.24.11.	input text Objekt	725
4.3.2.2.4.3.25.	label Objekt	729
4.3.2.2.4.3.26.	link Objekt	734
4.3.2.2.4.3.27.	document.links Collection des Internet Explorer (HTML-Element mit HREF-Attribut)	738
4.3.2.2.4.3.28.	map Objekt des Internet Explorer	738
4.3.2.2.4.3.29.	map.areas Collection des Internet Explorer	742
4.3.2.2.4.3.30.	marquee Objekt des Internet Explorer	742
4.3.2.2.4.3.31.	meta Objekt des Internet Explorer	748
4.3.2.2.4.3.32.	noFrames Objekt des Internet Explorer	750
4.3.2.2.4.3.33.	noScript Objekt des Internet Explorer	751
4.3.2.2.4.3.34.	object Objekt des Internet Explorer	752
4.3.2.2.4.3.35.	document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)	756
4.3.2.2.4.3.36.	document.select Objekt des Internet Explorer	757
4.3.2.2.4.3.36.1.	document.select.option Objekt des Internet Explorer	763
4.3.2.2.4.3.36.2.	document.select.options Collection des Internet Explorer	769
4.3.2.2.4.3.37.	document.selection Objekt des Internet Explorer	770
4.3.2.2.4.3.37.1.	document.selection.controlrange Collection des Internet Explorer	770
4.3.2.2.4.3.37.2.	document.selection.textrange Collection des Internet Explorer	771
4.3.2.2.4.3.38.	span Objekt	771
4.3.2.2.4.3.39.	style Objekt des Internet Explorer	778
4.3.2.2.4.3.39.1.	Style-Sheet-Deklarationen	781
4.3.2.2.4.3.39.2.	Style-Sheet und vordefinierte Werte	784
4.3.2.2.4.3.39.3.	Style-Sheet aus Datei einbinden	785
4.3.2.2.4.3.39.4.	Style-Sheet und Wertzuweisung an Eigenschaften	786
4.3.2.2.4.3.39.5.	Style-Sheet und Ausgabemedien (Pseudoklasse @media)	786
4.3.2.2.4.3.39.6.	Style-Sheet und Seiten-Eigenschaften (Pseudoklasse (@page)	787
4.3.2.2.4.3.39.7.	Style-Sheet und HTML-Tag-bezogene Eigenschaften	787
4.3.2.2.4.3.39.8.	Style-Sheet-Beispiele	796
4.3.2.2.4.3.39.9.	CSS-Konformität der Versionen Internet Explorer	801
4.3.2.2.4.3.39.9.1.	CSS1-Konformität des Internet Explorer 6.x zu seinen Vorgängern	801
4.3.2.2.4.3.39.9.2.	Arten der vollen CSS1 - Konformität des Internet Explorer ab 6.x	802
4.3.2.2.4.3.39.9.3.	CSS und Attribute width und height	803
4.3.2.2.4.3.39.10.	Kommentare innerhalb <STYLE> <STYLE>	804
4.3.2.2.4.3.39.11.	Kodierung von Werten mit erlaubter Einheit	805
4.3.2.2.4.3.39.12.	Kodierung von CSS-Werten in Style-Sheets (Klammerung in " " bzw. ' ')	805
4.3.2.2.4.3.39.13.	RGB-Farbangaben	805
4.3.2.2.4.3.39.14.	CLASS-Attribut und ID-Attribut-Wert	805
4.3.2.2.4.3.39.15.	Verwendung von \	805
4.3.2.2.4.3.39.16.	Die Eigenschaften des Internet Explorer als Grafiken	806
4.3.2.2.4.3.39.17.	Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung	807
4.3.2.2.4.3.39.18.	Dynamischen Eigenschaftenveränderung zur Laufzeit	807
4.3.2.2.4.3.39.19.	Eigenschaften	807
4.3.2.2.4.3.39.19.1.	Eigenschaften komplett	808
4.3.2.2.4.3.39.19.2.	Style-Eigenschaften, die nur per Script ansprechbar sind	816
4.3.2.2.4.3.39.19.3.	Style-Eigenschaften, die nur per STYLE-Attribut ansprechbar sind	818
4.3.2.2.4.3.39.19.4.	Style-Eigenschaften, die nur im HEAD des Dokumentes ansprechbar sind	818
4.3.2.2.4.3.39.19.5.	Style-Eigenschaftenübersicht zu Script- und STYLE-Kodierung	818
4.3.2.2.4.3.39.20.	Methoden	826
4.3.2.2.4.3.39.20.1.	Methoden des DOM	826
4.3.2.2.4.3.39.20.2.	Konvertierungsmethode	828



4.3.2.2.4.3.39.21.	Objekte (Auswahl) und ihr Style-Eigenschaften	828
4.3.2.2.4.3.39.22.	Verwaltung des Styles im Dokument	847
4.3.2.2.4.3.39.22.1.	currentStyle Objekt des Internet Explorer	847
4.3.2.2.4.3.39.22.2.	runtimeStyle Objekt des Internet Explorer	849
4.3.2.2.4.3.39.23.	Standard-Behavior (Verhaltensweisen) von Objekten des Internet Explorer (Auswahl)	850
4.3.2.2.4.3.39.23.1.	.style.clientCaps Behavior des Internet Explorer	852
4.3.2.2.4.3.39.23.2.	.style.download Behavior des Internet Explorer	855
4.3.2.2.4.3.39.23.3.	.style.homePage Behavior des Internet Explorer	856
4.3.2.2.4.3.39.23.4.	.style.httpFolder Behavior des Internet Explorer	857
4.3.2.2.4.3.39.23.5.	.style.mediaBar Behavior des Internet Explorer	858
4.3.2.2.4.3.39.23.6.	.style.saveFavorite Behavior des Internet Explorer	862
4.3.2.2.4.3.39.23.7.	.style.saveHistory Behavior des Internet Explorer	863
4.3.2.2.4.3.39.23.8.	.style.saveSnapshot Behavior des Internet Explorer	864
4.3.2.2.4.3.39.23.9.	.style.time2 Behavior des Internet Explorer	865
4.3.2.2.4.3.39.23.9.1.	Timer Konzept des Internet Explorer	865
4.3.2.2.4.3.39.23.9.1.1.	Ansatz	865
4.3.2.2.4.3.39.23.9.1.2.	Alternative zum Timer Konzept des IE	866
4.3.2.2.4.3.39.23.9.1.3.	XML-Kodierung in HTML	866
4.3.2.2.4.3.39.23.9.1.4.	Timeline für ein Objekt anhand HTML-Attribute erzeugen (BEGIN, END, DUR, TIMEACTION)	867
4.3.2.2.4.3.39.23.9.1.5.	Timeline für ein Objektgruppe (Timecontainer) erzeugen (Attribut TIMECONTAINER)	868
4.3.2.2.4.3.39.23.9.1.5.1.	Timeline für eine Objektgruppe aus einem unverschachtelten Timer	869
4.3.2.2.4.3.39.23.9.1.5.2.	Timeline für eine Objektgruppe aus verschachtelten Timern	870
4.3.2.2.4.3.39.23.9.1.6.	Time Formate zum .style.time2 Behavior des Internet Explorer	872
4.3.2.2.4.3.39.23.9.1.7.	Alternativen zum Behavior .style.time2	873
4.3.2.2.4.3.39.23.9.1.8.	Medien zur Animation per Behavior .style.time2	874
4.3.2.2.4.3.39.23.9.2.	Beschreibung des Behavior	875
4.3.2.2.4.3.39.23.9.2.1.	Syntax	875
4.3.2.2.4.3.39.23.9.2.2.	Kodierung des .style.time2 Behavior in HTML	876
4.3.2.2.4.3.39.23.9.2.3.	Import des Behavior .style.time2 in HTML	876
4.3.2.2.4.3.39.23.9.2.4.	Bezug des Timers auf ein Element (Objekt) in HTML	876
4.3.2.2.4.3.39.23.9.2.4.1.	Bezug auf ein Element, das nicht Media-Daten enthält	877
4.3.2.2.4.3.39.23.9.2.4.2.	Bezug auf ein Element, das nur Media-Daten enthält	877
4.3.2.2.4.3.39.23.9.2.4.3.	Bezug auf ein Element ohne Media-Daten, das gemeinsam mit Media-Daten animiert werden soll	878
4.3.2.2.4.3.39.23.9.2.5.	HTML-Style-Attribut im zu animierenden Element	878
4.3.2.2.4.3.39.23.9.2.6.	Animation eines Elementes anhand einer speziellen Eigenschaft im HTML-Attribut STYLE	878
4.3.2.2.4.3.39.23.9.2.7.	Implementierung Timer bzw. Neusetzung seiner Attribute bei aktivem / nichtaktivem Element	878
4.3.2.2.4.3.39.23.9.2.8.	Synchronisierung von Timelines	878
4.3.2.2.4.3.39.23.9.2.8.1.	Synchronisierung von verschachtelten Timelines (verschachtelte Timer)	878
4.3.2.2.4.3.39.23.9.2.8.2.	Zwangs-Synchronisierung von Timelines	879
4.3.2.2.4.3.39.23.9.2.8.3.	Synchronisation von Eltern-Time-Container mit seinen Elementen	881
4.3.2.2.4.3.39.23.9.2.9.	Beispiele in HTML	882
4.3.2.2.4.3.39.23.9.2.9.	Eigenschaften	885
4.3.2.2.4.3.39.23.9.2.10.	Methoden	885
4.3.2.2.4.3.39.23.9.2.11.	Events	887
4.3.2.2.4.3.39.23.9.2.12.	Objekte und Collectionen zur Verwaltung von style.time2	887
4.3.2.2.4.3.39.23.9.2.12.1.	currTimeState Objekt des Internet Explorer	887
4.3.2.2.4.3.39.23.9.2.12.2.	timeChildren Collection des Internet Explorer	890
4.3.2.2.4.3.39.23.9.3.	Behavior-Objekte und Behavior-Collectionen	890
4.3.2.2.4.3.39.23.9.3.1.	.style.time2.animate Behavior-Objekt des Internet Explorer	890
4.3.2.2.4.3.39.23.9.3.2.	.style.time2.animateColor Behavior-Objekt des Internet Explorer	896
4.3.2.2.4.3.39.23.9.3.3.	.style.time2.animateMotion Behavior-Objekt des Internet Explorer	900
4.3.2.2.4.3.39.23.9.3.4.	.style.time2.animation Behavior-Objekt des Internet Explorer	907
4.3.2.2.4.3.39.23.9.3.5.	.style.time2.audio Behavior-Objekt des Internet Explorer	911
4.3.2.2.4.3.39.23.9.3.6.	.style.time2.excl Behavior-Objekt des Internet Explorer	917
4.3.2.2.4.3.39.23.9.3.7.	.style.time2.img Behavior-Objekt des Internet Explorer	920
4.3.2.2.4.3.39.23.9.3.8.	.style.time2.media Behavior-Objekt des Internet Explorer	924
4.3.2.2.4.3.39.23.9.3.9.	.style.time2.par Behavior-Objekt des Internet Explorer	930
4.3.2.2.4.3.39.23.9.3.10.	.style.time2.playItem Behavior-Objekt des Internet Explorer	934
4.3.2.2.4.3.39.23.9.3.11.	.style.time2.playList Behavior-Collection des Internet Explorer	937
4.3.2.2.4.3.39.23.9.3.12.	.style.time2.priorityClass Behavior-Objekt des Internet Explorer	940
4.3.2.2.4.3.39.23.9.3.13.	.style.time2.ref Behavior-Objekt des Internet Explorer	944
4.3.2.2.4.3.39.23.9.3.14.	.style.time2.set Behavior-Objekt des Internet Explorer	949
4.3.2.2.4.3.39.23.9.3.15.	.style.time2.seq Behavior-Objekt des Internet Explorer	952
4.3.2.2.4.3.39.23.9.3.16.	.style.time2.switch Behavior-Objekt des Internet Explorer	957
4.3.2.2.4.3.39.23.9.3.17.	.style.time2.transitionFilter Behavior-Objekt des Internet Explorer	959
4.3.2.2.4.3.39.23.9.3.18.	.style.time2.video Behavior-Objekt des Internet Explorer	967
4.3.2.2.4.3.39.24.	Userdaten verwalten (.style.userData Behavior des Internet Explorer)	974
4.3.2.2.4.3.39.25.	HTML-Dokument mit Style-Sheet (CSS) im IE und NS	976
4.3.2.2.4.3.39.25.1.	Style-Sheet-Deklarations-Varianten	977
4.3.2.2.4.3.39.25.1.1.	Style-Sheet Eigenschaften innerhalb <HEAD> deklarieren	977
4.3.2.2.4.3.39.25.1.2.	Style-Sheet-Format und Style-Sheet-Eigenschaften als tag-unabhängig deklarieren (Attribut ID)	977



4.3.2.2.4.3.39.25.1.3.	Style-Sheet-Schlüsselwort und Style-Sheet-Eigenschaften als tag-abhängig deklarieren	977
4.3.2.2.4.3.39.25.1.4.	Style-Sheet Eigenschaften ohne Tag-Attribut ID deklarieren	978
4.3.2.2.4.3.39.25.1.5.	Style-Sheet-Unterklasse deklarieren	978
4.3.2.2.4.3.39.25.1.6.	Style-Sheet-Eigenschaften mit Attribut STYLE deklarieren	978
4.3.2.2.4.3.39.25.1.7.	Style-Sheet-Eigenschaften eines HTML-Elementes deklarieren	979
4.3.2.2.4.3.39.25.1.8.	Beispiele für Style-Sheet	979
4.3.2.2.4.3.39.25.2.	Style-Sheet und numerische (nicht vordefinierte) Farbangaben	980
4.3.2.2.4.3.39.25.3.	Style-Sheet und vordefinierte Bezeichner	980
4.3.2.2.4.3.39.25.3.1.	Style-Sheet und vordefinierte Dimensionen	980
4.3.2.2.4.3.39.25.3.2.	Style-Sheet- und Dezimalkomma	980
4.3.2.2.4.3.39.25.3.3.	Style-Sheet und vordefinierte Schriften (Font und Style)	980
4.3.2.2.4.3.39.25.3.4.	Style-Sheet und vordefinierte Farbbereiche	981
4.3.2.2.4.3.39.25.3.4.1.	Style-Sheet und Farbe des Desktop	981
4.3.2.2.4.3.39.25.3.4.2.	Style-Sheet und Farbe des Dokumentfensters	981
4.3.2.2.4.3.39.25.3.4.3.	Style-Sheet und Farbe aktives Fenster	981
4.3.2.2.4.3.39.25.3.4.4.	Style-Sheet und Farbe inaktives Fenster	981
4.3.2.2.4.3.39.25.3.4.5.	Style-Sheet und Farbe des Dialogfensters	981
4.3.2.2.4.3.39.25.3.4.6.	Style-Sheet und Farbe einer Auswahlliste	981
4.3.2.2.4.3.39.25.3.4.7.	Style-Sheet und Farbe von Tooltip und Popuphilfe (Hint)	981
4.3.2.2.4.3.39.25.3.4.8.	Style-Sheet und Farbe eines Menü	981
4.3.2.2.4.3.39.25.3.4.9.	Style-Sheet und Farbe der Scrollleiste	981
4.3.2.2.4.3.39.25.3.4.10.	Style-Sheet und Farbe eines 3D-Elements	981
4.3.2.2.4.3.39.25.4.	Style-Sheet-Dateien	981
4.3.2.2.4.3.39.25.4.1.	Style-Sheet-Schriftdatei laden (*.eot bzw. *.prf)	981
4.3.2.2.4.3.39.25.4.2.	Style-Sheet-Informationen aus Datei einbinden (*.css)	981
4.3.2.2.4.3.39.25.4.3.1.	Ausgabemedien-spezifische CSS-Datei importieren (@import)	982
4.3.2.2.4.3.39.25.4.3.2.	Ausgabemedien-spezifische Style-Sheet-Eigenschaften deklarieren (@media)	982
4.3.2.2.4.3.39.25.4.4.	Style-Sheet und Font-Dateien (@font-face) anhand browserinterner Pseudoklasse	982
4.3.2.2.4.3.39.25.4.5.	Style-Sheet und Seiten-Eigenschaften (@page) anhand browserinterner Pseudoklasse	982
4.3.2.2.4.3.39.25.4.6.	Style-Sheet und Hintergrund-Grafik	983
4.3.2.2.4.3.39.25.4.7.	Style-Sheet und Rahmen-Eigenschaften (Border)	983
4.3.2.2.4.3.39.25.4.8.	Style-Sheet und HTML-Tag-bezogene Eigenschaften	984
4.3.2.2.4.3.39.25.4.8.1.	Style-Sheet-Eigenschaften zu <A>	984
4.3.2.2.4.3.39.25.4.8.2.	Style-Sheet-Eigenschaften zu <P>	984
4.3.2.2.4.3.39.25.4.8.3.	Style-Sheet-Eigenschaften zu <H1> bis <H6>	984
4.3.2.2.4.3.39.25.5.	Style-Sheet-Eigenschaften (Style-Sheet-Formatangaben)	984
4.3.2.2.4.3.39.25.5.1.	Style-Sheet und Wertzuweisung an Eigenschaften	984
4.3.2.2.4.3.39.25.5.2.	Style-Sheet-Eigenschaften für HTML-Elemente	984
4.3.2.2.4.3.39.25.5.2.1.	Style-Sheet und Auswertung von Pixelpositions-Angaben	984
4.3.2.2.4.3.39.25.5.2.2.	Style-Sheet und Abstand von HTML-Elementen	984
4.3.2.2.4.3.39.25.5.2.3.	Style-Sheet und HTML-Element-Dimension (-Grenzen, -Anzeigebereich)	984
4.3.2.2.4.3.39.25.5.2.4.	Style-Sheet und HTML-Element-Sichtbarkeit	985
4.3.2.2.4.3.39.25.5.2.5.	Style-Sheet und Element-Ausschnitt	985
4.3.2.2.4.3.39.25.5.2.6.	Style-Sheet und Element-Scrolling	985
4.3.2.2.4.3.39.25.5.2.7.	Style-Sheet und Layer-Element	985
4.3.2.2.4.3.39.25.5.2.8.	Style-Sheet und HTML-Elemente-Anordnung im Dokument	985
4.3.2.2.4.3.39.25.5.2.9.	Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet)	985
4.3.2.2.4.3.39.25.5.3.	Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung)	985
4.3.2.2.4.3.39.25.5.4.	Style-Sheet-Eigenschaften für Tabelle	985
4.3.2.2.4.3.39.25.5.5.	Style-Sheet-Eigenschaften für Seitendarstellung im Dokument	986
4.3.2.2.4.3.39.25.5.6.	Style-Sheet-Eigenschaften für Text	986
4.3.2.2.4.3.39.25.5.7.	Style-Sheet-Eigenschaften für Font	987
4.3.2.2.4.3.39.25.5.8.	Style-Sheet-Eigenschaften für Mauscursor	988
4.3.2.2.4.3.39.25.5.9.	Style-Sheet-Eigenschaften für Unicode (Zeichensatz)	988
4.3.2.2.4.3.39.25.6.	Style-Sheet-Beispiele	988
4.3.2.2.4.3.39.26.	Beispiele für Objekteigenschaft .style	989
4.3.2.2.4.3.39.26.1.	Zeichensatz- und Texteeigenschaften	989
4.3.2.2.4.3.39.26.2.	Farben- und Hintergrundeeigenschaften	990
4.3.2.2.4.3.39.26.3.	Layouteeigenschaften	990
4.3.2.2.4.3.39.26.4.	Positionierungsangaben	992
4.3.2.2.4.3.39.26.5.	Druckeeigenschaften	993
4.3.2.2.4.3.39.26.6.	Cursor	993
4.3.2.2.4.3.39.26.7.	Bildausschnitt	993
4.3.2.2.4.3.40.	styleSheet Objekt des Internet Explorer	994
4.3.2.2.4.3.40.1.	styleSheet.imports Collection des Internet Explorer	997
4.3.2.2.4.3.40.2.	page Objekt des Internet Explorer	998
4.3.2.2.4.3.40.3.	styleSheet.pages Collection des Internet Explorer	998
4.3.2.2.4.3.40.4.	styleSheet.rules Collection des Internet Explorer	998
4.3.2.2.4.3.41.	document.styleSheets Collection des Internet Explorer	999
4.3.2.2.4.3.42.	table Objekt des Internet Explorer	1000
4.3.2.2.4.3.42.1.	Erzeugung der Tabelle	1000



4.3.2.2.4.3.42.1.1.	Erzeugung in HTML	1000	
4.3.2.2.4.3.42.1.2.	Erzeugung in JScript	1001	
4.3.2.2.4.3.42.2.	Daten der Tabelle	1001	
4.3.2.2.4.3.42.2.1.	Datenbereitstellung	1001	
4.3.2.2.4.3.42.2.1.1.	in HTML	1001	
4.3.2.2.4.3.42.2.1.2.	in JScript	1001	
4.3.2.2.4.3.42.2.1.3.	per Active-X-Control	1001	
4.3.2.2.4.3.42.2.2.	Dateneinbindung	1001	
4.3.2.2.4.3.42.2.2.1.	in HTML	1001	
4.3.2.2.4.3.42.2.2.2.	in JScript	1001	
4.3.2.2.4.3.42.2.2.3.	per Active-X-Control	1001	
4.3.2.2.4.3.42.3.	Tabellen-Objektmodell (TOM) in JScript	1001	
4.3.2.2.4.3.42.4.	Methoden des DOM und TOM zur Verwaltung der Tabellenelemente (Übersicht)	1001	
4.3.2.2.4.3.42.5.	Dynamische Struktur und Daten einer Tabelle per JScript	1002	
4.3.2.2.4.3.42.5.1.	Strukturerzeugung der Tabelle	1002	
4.3.2.2.4.3.42.5.1.1.	in HTML	1002	
4.3.2.2.4.3.42.5.1.2.	per Methoden des DOM	1004	
4.3.2.2.4.3.42.5.2.	Datenerzeugung mit der Strukturbildung und zur Laufzeit	1006	
4.3.2.2.4.3.42.6.	Dynamische Veränderung einer Tabelle in JScript	1014	
4.3.2.2.4.3.42.6.1.	Datenveränderung per JScript	1014	
4.3.2.2.4.3.42.6.2.	Layoutveränderung per Style	1014	
4.3.2.2.4.3.42.6.3.	Strukturveränderung per JScript	1014	
4.3.2.2.4.3.42.7.	Eigenschaften der Tabelle	1014	
4.3.2.2.4.3.42.8.	Methoden der Tabelle	1018	
4.3.2.2.4.3.42.9.	table.caption Objekt des Internet Explorer	1027	
4.3.2.2.4.3.42.10.	table.col Objekt des Internet Explorer	1032	
4.3.2.2.4.3.42.11.	table.colGroup Objekt des Internet Explorer	1035	
4.3.2.2.4.3.42.12.	table.rows Collection des Internet Explorer	1039	
4.3.2.2.4.3.42.13.	table.rows.cells Collection des Internet Explorer	1040	
4.3.2.2.4.3.42.14.	table.tBody Objekt des Internet Explorer	1040	
4.3.2.2.4.3.42.14.1.	table.tBody.rows Collection des Internet Explorer	1045	
4.3.2.2.4.3.42.14.2.	table.tBody.rows.cells Collection des Internet Explorer	1045	
4.3.2.2.4.3.42.15.	table.tBodies Collection des Internet Explorer	1046	
4.3.2.2.4.3.42.16.	table.tFoot Objekt des Internet Explorer	1046	
4.3.2.2.4.3.42.16.1.	table.tFoot.rows Collection des Internet Explorer	1051	
4.3.2.2.4.3.42.16.2.	table.tFoot.rows.cells Collection des Internet Explorer	1051	
4.3.2.2.4.3.42.17.	table.tHead Objekt des Internet Explorer	1051	
4.3.2.2.4.3.42.17.1.	table.tHead.rows Collection des Internet Explorer	1056	
4.3.2.2.4.3.42.17.2.	table.tHead.rows.cells Collection des Internet Explorer	1056	
4.3.2.2.4.3.42.18.	table.tr Objekt des Internet Explorer	1056	
4.3.2.2.4.3.42.18.1.	table.tr.td Objekt des Internet Explorer	1062	
4.3.2.2.4.3.42.18.2.	table.tr.th Objekt des Internet Explorer	1068	
4.3.2.2.4.3.43.	textarea Objekt des Internet Explorer	1074	
4.3.2.2.4.3.44.	document.TextRange Objekt des Internet Explorer (Textbereich im Dokument)	1082	
4.3.2.2.4.3.44.1.	document.TextRange.TextRectangle Collection des Internet Explorer	1085	
4.3.2.2.4.3.44.2.	document.TextRange.TextRectangle Objekt des Internet Explorer	1085	
4.3.2.2.5.	window.event Objekt	1087	
4.3.2.2.5.1.	Eventarten Internet Explorer und Netscape	1095	
4.3.2.2.5.1.1.	Eventarten der HTML-Tags (Auswahl)	1095	
4.3.2.2.5.1.2.	Eventarten des IE und NS (Auswahl)	1097	
4.3.2.2.5.2.	Eigenschaften im IE und NS (Auswahl)	1104	
4.3.2.2.5.3.	Methoden im IE und NS	1106	
4.3.2.2.5.4.	event Objekt des Internet Explorer	1106	
4.3.2.2.5.4.1.	Zugriff	1106	
4.3.2.2.5.4.1.	Eigenschaften	1106	
4.3.2.2.5.4.2.	Methoden	1107	
4.3.2.2.5.4.3.	event.bookmarks Collection	1108	
4.3.2.2.5.4.4.	event.dataTransfer Objekt des Internet Explorer	1108	
4.3.2.2.5.4.5.	Eventarten (Auswahl)	1112	
4.3.2.2.5.4.6.	Eventarten wichtiger Objekte (Auswahl)	1131	
4.3.2.2.5.5.	Event-Behandlung beim Internet Explorer bzw. Netscape	1142	
4.3.2.2.5.5.1.	Ansatz	1142	
4.3.2.2.5.5.2.	Ereignis und Eventhandler	1143	
4.3.2.2.5.5.3.	Eventbehandlung im Netscape	1144	
4.3.2.2.5.5.3.1.	Event des Netscape einer Nicht-Standardbehandlung unterziehen	1144	
4.3.2.2.5.5.3.1.1.	Event und Eventhandler dem Objekt window zuordnen (captureEvents(event_liste))	1144	
4.3.2.2.5.5.3.1.2.	Event entlang der Eventhierarchie weiterreichen	1144	
4.3.2.2.5.5.3.1.2.1.	Event entlang der Nicht-Standard- Eventhierarchie weiterreichen (handleEvent(event_objekt))	1145	
4.3.2.2.5.5.3.1.2.2.	Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))	1145	
4.3.2.2.5.5.3.2.	Event des Netscape von Nicht-Standardbehandlung wieder der	1145	
4.3.2.2.5.5.3.2.1.	Standard-Eventhandler dem Objekt window zuordnen (releaseEvents(event_liste))	1145	



4.3.2.2.5.3.2.2.	Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))	1145
4.3.2.2.5.3.3.	Eventbehandlung durch eine Fremdseite mit signiertem Script	1145
4.3.2.2.5.3.3.1.	Beschaffung der Rechte "UniversalBrowserWrite" bzw. "UniversalBrowserWrite"	1145
4.3.2.2.5.3.3.2.	Eventbehandlung durch Fremde Seite aktivieren (enableExternalCapture())	1146
4.3.2.2.5.3.3.3.	Eventbehandlung durch Fremde Seite deaktivieren (disableExternalCapture())	1146
4.3.2.2.5.3.4.	Beispiel	1146
4.3.2.2.5.5.4.	Eventbehandlung beim Internet Explorer	1147
4.3.2.2.5.5.4.1.	Event des IE einer Nicht-Standardbehandlung unterziehen	1147
4.3.2.2.5.5.4.2.	Event von Nicht-Standardbehandlung wieder der Standardbehandlung	1147
4.3.2.2.5.5.4.3.	Ereignisbehandlung für mouseover und mouseout ein-bzw. ausschalten	1147
4.3.2.2.5.5.4.4.	Event bei Behavior (Verhaltensweise)	1147
4.3.2.2.5.5.5.	Fehlerbehandlung per onerror	1147
4.3.2.2.5.5.5.1.	onerror-Standard abschalten	1147
4.3.2.2.5.5.5.2.	onerror-Routine privater Art (eigene Fehlerbehandlung einrichten)	1147
4.3.2.2.5.5.6.	Eventbehandlung in einem Formular des IE und NS	1149
4.3.2.2.5.5.7.	Eventbehandlung bei Drag & Drop	1150
4.3.2.2.5.5.7.1.	Eventbehandlung bei Drag & Drop eines HTML-Elementes beim Internet Explorer	1150
4.3.2.2.5.5.7.1.1.	Eventarten für Drag & Drop	1150
4.3.2.2.5.5.7.1.2.	Beispiel für Drag & Drop	1151
4.3.2.2.5.5.7.2.	Eventbehandlung bei Drag & Drop von Dateien und Verknüpfungen beim Netscape ab 4.x	1152
4.3.2.2.5.5.8.	Tastatur-Eventbehandlung des IE und NS	1152
4.3.2.2.5.5.8.1.	Tastatur-Eventbehandlung beim Internet Explorer	1152
4.3.2.2.5.5.8.1.1.	Tastatur-Eventarten	1152
4.3.2.2.5.5.8.1.2.	Tastatur-Eventeigenschaften	1153
4.3.2.2.5.5.8.1.2.1.	Alle Tasten (.keyCode und .repeat)	1153
4.3.2.2.5.5.8.1.2.2.	Steuertasten Alt, Strg (Ctrl) und Umschalt (Shift) (.xxxKey und .xxxLeft)	1153
4.3.2.2.5.5.8.1.2.3.	Eventhandler-Eigenschaften (.returnValue)	1153
4.3.2.2.5.5.8.1.2.4.	HTML-Element-Event-Eigenschaft (.srcElement)	1153
4.3.2.2.5.5.8.1.2.5.	Eventart-Eigenschaft (.type)	1153
4.3.2.2.5.5.8.1.3.	Tastatur-Ereignis-Folge onkeydown und onkeypress	1153
4.3.2.2.5.5.8.1.3.1.	Tastatur-Ereignis onkeydown	1153
4.3.2.2.5.5.8.1.3.2.	Tastatur-Ereignis onkeypress	1154
4.3.2.2.5.5.8.2.	Tastatur-Eventbehandlung beim Netscape	1155
4.3.2.2.5.5.8.2.1.	Tastatur-Eventarten	1155
4.3.2.2.5.5.8.2.2.	Tastatur-Eventeigenschaften	1155
4.3.2.2.5.5.8.2.2.1.	Steuertasten Alt, Strg (Ctrl) und Umschalt (Shift) (.modifiers)	1155
4.3.2.2.5.5.8.2.2.2.	Eventwert-Eigenschaft (.which)	1156
4.3.2.2.5.5.8.2.2.3.	Eventart-Eigenschaft (.type)	1156
4.3.2.2.5.5.8.2.2.4.	Eventquelle-Eigenschaft (.target)	1156
4.3.2.2.5.5.8.2.3.	Tastatur-Ereignis-Folge onkeydown und onkeypress	1156
4.3.2.2.5.5.8.2.3.1.	Tastatur-Ereignis onkeydown	1156
4.3.2.2.5.5.8.2.3.2.	Tastatur-Ereignis onkeypress	1156
4.3.2.2.5.5.8.3.	Beispiel zur Tastatur-Eventbehandlung beim IE und Netscape	1156
4.3.2.2.5.5.9.	Mouse-Eventbehandlung des IE und NS	1157
4.3.2.2.5.5.9.1.	Mouse-Eventbehandlung beim Internet Explorer ab 4.x	1157
4.3.2.2.5.5.9.1.1.	Mouse-Eventarten	1157
4.3.2.2.5.5.9.1.2.	Mouse-Event-Eigenschaften	1158
4.3.2.2.5.5.9.2.	Mouse-Eventbehandlung beim Netscape ab 4.x	1159
4.3.2.2.5.5.9.2.1.	Mouse-Eventarten	1159
4.3.2.2.5.5.9.2.2.	Mouse-Event-Eigenschaften	1159
4.3.2.2.5.5.10.	Eventbehandlung für Textoperationen mit der Windows-Zwischenablage (Clipboard)	1160
4.3.2.2.5.5.10.1.	Eventarten	1160
4.3.2.2.5.5.10.2.	Beispiel	1160
4.3.2.2.5.5.11.	Druck-Eventbehandlung nur Internet Explorer ab 5.x	1161
4.3.2.2.5.5.12.	HTML-Element-Lade-Eventbehandlung des IE und NS	1161
4.3.2.2.5.5.12.1.	Lade-Ereignisse des Internet Explorer ab 4.x bzw. 5.x	1161
4.3.2.2.5.5.12.1.1.	Lade-Ereignisse für Bild	1161
4.3.2.2.5.5.12.1.2.	Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)	1161
4.3.2.2.5.5.12.2.	Lade-Ereignisse des Netscape ab 3.x	1161
4.3.2.2.5.5.12.2.1.	Lade-Ereignisse für Bild	1161
4.3.2.2.5.5.12.2.2.	Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)	1162
4.3.2.2.6.	window.external Objekt des Internet Explorer	1162
4.3.2.2.7.	window.history Objekt (history Collection)	1165
4.3.2.2.8.	window.location Objekt	1166
4.3.2.2.9.	window.navigator Objekt	1167
4.3.2.2.9.1.	navigator Objekt im Netscape	1167
4.3.2.2.9.1.1.	Eigenschaften	1167
4.3.2.2.9.1.2.	Methoden	1168
4.3.2.2.9.1.3.	navigator.plugins Collection des Netscape	1168
4.3.2.2.9.1.4.	navigator.mimeTypes Collection	1169
4.3.2.2.9.2.	navigator Objekt im Internet Explorer	1169



4.3.2.2.9.2.1.	Eigenschaften	1169	
4.3.2.2.9.2.2.	Methoden	1170	
4.3.2.2.9.2.3.	navigator.mimeTypes Collection (auch bei IE 6.x)	1170	
4.3.2.2.9.2.4.	navigator.plugins Collection des Internet Explorer (auch bei IE 6.x)	1171	
4.3.2.2.9.2.5.	navigator.userProfile Objekt des Internet Explorer	1171	
4.3.2.2.10.	window.popup Objekt des Internet Explorer	1172	
4.3.2.2.11.	window.screen Objekt	1182	
4.3.2.2.11.1.	screen Objekt im Netscape	1182	
4.3.2.2.11.2.	screen Objekt im Internet Explorer	1182	
4.3.2.2.12.	Sonstige Objekte und Collectionen (Auswahl)	1183	
4.3.2.2.12.1.	regexp Objekt als Instanz des Script-Objektes RegExp	1183	
4.3.2.2.12.2.	RegExp Objekt (nicht regexp Objekt)	1186	
4.3.2.2.12.3.	script Objekt des Internet Explorer	1187	
4.3.2.2.12.4.	document.scripts Collection des Internet Explorer	1190	
4.3.2.2.12.5.	var Objekt des Internet Explorer	1190	
4.3.2.2.12.6.	xml Objekt im Internet Explorer	1195	
4.3.2.2.12.7.	Collectionen des Internet Explorers - Übersicht (englisch)	1196	
4.3.2.2.12.7.1.	Collection .all	1196	
4.3.2.2.12.7.2.	Collection .anchors	1197	
4.3.2.2.12.7.3.	Collection .applets	1197	
4.3.2.2.12.7.4.	Collection .areas	1198	
4.3.2.2.12.7.5.	Collection .attributes	1198	
4.3.2.2.12.7.6.	Collection .behaviorUrns	1198	
4.3.2.2.12.7.7.	Collection .blockFormats	1199	
4.3.2.2.12.7.8.	Collection .boundElements	1199	
4.3.2.2.12.7.9.	Collection .cells	1199	
4.3.2.2.12.7.10.	Collection .childNodes	1199	
4.3.2.2.12.7.11.	Collection .children	1200	
4.3.2.2.12.7.12.	Collection .controlRange	1200	
4.3.2.2.12.7.13.	Collection .document.all	1201	
4.3.2.2.12.7.14.	Collection .elements	1201	
4.3.2.2.12.7.15.	Collection .embeds	1201	
4.3.2.2.12.7.16.	Collection .filters	1201	
4.3.2.2.12.7.17.	Collection .fonts	1202	
4.3.2.2.12.7.18.	Collection .forms	1202	
4.3.2.2.12.7.19.	Collection .frames	1202	
4.3.2.2.12.7.20.	Collection .images	1203	
4.3.2.2.12.7.21.	Collection .imports	1203	
4.3.2.2.12.7.22.	Collection .links	1203	
4.3.2.2.12.7.23.	Collection .namespaces	1203	
4.3.2.2.12.7.24.	Collection .options	1204	
4.3.2.2.12.7.25.	Collection .pages	1204	
4.3.2.2.12.7.26.	Collection .plugins	1204	
4.3.2.2.12.7.27.	Collection .rows	1204	
4.3.2.2.12.7.28.	Collection .rules	1205	
4.3.2.2.12.7.29.	Collection .scripts	1205	
4.3.2.2.12.7.30.	Collection .styleSheets	1205	
4.3.2.2.12.7.31.	Collection .tBodies	1206	
4.3.2.2.12.7.32.	Collection .TextRange	1206	
4.3.2.2.12.7.33.	Collection .TextRectangle	1206	
4.3.2.2.12.8.	wichtige Events und ihre Objekte - Übersicht (z.T. in englisch), Methode .fireEvent()	1207	
4.3.2.2.12.9.	wichtige Objekte - Übersicht (z.T. englisch)	1221	
4.3.2.2.12.9.1.	A	1224	
4.3.2.2.12.9.2.	BGSOUND	1229	
4.3.2.2.12.9.3.	BODY	1231	
4.3.2.2.12.9.4.	clientInformation	1235	
4.3.2.2.12.9.5.	DIV	1236	
4.3.2.2.12.9.6.	document	1240	
4.3.2.2.12.9.7.	event	1242	
4.3.2.2.12.9.8.	FONT	1245	
4.3.2.2.12.9.9.	FRAME	1248	
4.3.2.2.12.9.10.	FRAMESET	1250	
4.3.2.2.12.9.11.	HEAD	1251	
4.3.2.2.12.9.12.	HTML	1253	
4.3.2.2.12.9.13.	HTML-Kommentar	1255	
4.3.2.2.12.9.14.	IMG	1255	
4.3.2.2.12.9.15.	Input	1260	
4.3.2.2.12.9.16.	Input button	1261	
4.3.2.2.12.9.17.	Input checkbox	1264	
4.3.2.2.12.9.18.	Input file	1268	
4.3.2.2.12.9.19.	Input radio	1272	



4.3.2.2.12.9.20.	Input text	1276	
4.3.2.2.12.9.21.	LINK	1280	
4.3.2.2.12.9.22.	location	1281	
4.3.2.2.12.9.23.	MARQUEE	1281	
4.3.2.2.12.9.24.	navigator	1286	
4.3.2.2.12.9.25.	OBJECT	1286	
4.3.2.2.12.9.26.	OPTION	1299	
4.3.2.2.12.9.27.	P	1301	
4.3.2.2.12.9.28.	PARAM	1305	
4.3.2.2.12.9.29.	popup	1305	
4.3.2.2.12.9.30.	screen	1313	
4.3.2.2.12.9.31.	SCRIPT	1313	
4.3.2.2.12.9.32.	SELECT	1316	
4.3.2.2.12.9.33.	SPAN	1319	
4.3.2.2.12.9.34.	STYLE (STYLE-Attribut oder .style, nicht styleSheet)	1323	
4.3.2.2.12.9.35.	styleSheet (nicht STYLE-Attribut oder .style)	1331	
4.3.2.2.12.9.36.	TABLE	1332	
4.3.2.2.12.9.37.	CAPTION	1339	
4.3.2.2.12.9.38.	TD	1342	
4.3.2.2.12.9.39.	TH	1346	
4.3.2.2.12.9.40.	TR	1350	
4.3.2.2.12.9.41.	TEXTAREA	1353	
4.3.2.2.12.9.42.	TextRange	1357	
4.3.2.2.12.9.43.	TITLE im HEAD	1359	
4.3.2.2.12.9.44.	userProfile	1360	
4.3.2.2.12.9.45.	window	1360	
4.3.2.2.12.9.46.	XMLHttpRequest	1369	
4.3.2.2.12.10.	Vordefinierte Farbbezeichner	1373	
5.	Plugins des Netscape und ActiveX-Controls des Internet Explorers	1375	
5.1.	Plugins des Netscape für Browsererweiterungen durch Fremdanbieter	1375	
5.2.	ActiveX des Internet Explorer für Browsererweiterungen	1375	
5.2.1.	Datenbank im Internet Explorer ab 4.x	1376	
5.2.1.1.	Aufbau der Datenbank	1376	
5.2.1.2.	HTML-Einbindung	1377	
5.2.1.2.1.	Objekt-Deklaration	1377	
5.2.1.2.2.	Datenfeld-Deklaration	1377	
5.2.1.3.	Operationen mit der Datenbank	1378	
5.2.1.3.1.	Datenbank-Objekt	1378	
5.2.1.3.2.	Objekt der Satzselektion (recordset)	1379	
5.2.1.4.	Beispiele	1379	
5.2.1.4.1.	Sortierung	1379	
5.2.1.4.2.	Blättern in Datenbank	1380	
5.2.1.4.3.	Satzselektion mit Filter	1382	
5.2.2.	Direct Animation im Internet Explorer (Übersicht DA als DirectX-Komponente)	1382	
5.2.2.1.	DA-Bibliothek	1388	
5.2.2.2.	DA-Objekte vordefiniert (Auswahl)	1389	
5.2.2.2.1.	DA-Farben	1389	
5.2.2.2.1.1.	DA-Farbe vordefiniert	1389	
5.2.2.2.1.2.	DA-Füllfarbe	1390	
5.2.2.2.2.	DA-Linie	1390	
5.2.2.2.3.	DA-Event	1390	
5.2.2.2.4.	DA-Timer	1390	
5.2.2.2.5.	DA-Zahl mit numerischem Wert	1390	
5.2.2.2.6.	DA-Operator	1390	
5.2.2.3.	Kombination von DA-Objekten anhand von Beispielen	1390	
5.2.2.3.1.	2D-Objekte in der Ebene bzw. im Raum	1390	
5.2.2.3.1.1.	2D-Objekte ohne Rotation: Geometrische Objekte und Text in der Ebene	1390	
5.2.2.3.1.2.	2D-Objekte mit 2D- und 3D-Rotation auf Basis einer periodischen Sinus-Schwingung	1393	
5.2.2.3.2.	Sound	1397	
5.2.2.3.2.1.	Sound ohne Kanalsteuerung	1397	
5.2.2.3.2.2.	Sound mit Kanalsteuerung	1398	
5.2.2.3.3.	Farbe	1399	
5.2.2.3.4.	Text	1400	
5.2.2.3.4.1.	Text ohne Hintergrund	1400	
5.2.2.3.4.2.	Text mit Hintergrund	1401	
5.2.2.3.5.	Font	1401	
5.2.2.3.5.1.	Standardfont	1401	
5.2.2.3.5.2.	Windows-Font	1403	
5.2.2.3.6.	Grafik aus externer Bilddatei	1405	
5.2.2.3.6.1.	Grafikfolge	1405	
5.2.2.3.6.2.	Grafik scrollend	1406	



5.2.2.3.6.3.	Grafik rotierend	1408	
5.2.2.3.7.	Sequenz	1409	
5.2.2.3.7.1.	Sequenz mit zeitlicher Begrenzung	1409	
5.2.2.3.7.2.	Sequenz mit Wertbereich-Begrenzung	1410	
5.2.2.3.8.	Event	1411	
5.2.2.3.8.1.	Eventhandler ohne Erweiterung	1411	
5.2.2.3.8.2.	Eventhandler mit Erweiterung	1413	
5.2.2.3.8.3.	Eventfähigkeit eines DA-Objektes	1415	
5.2.2.3.9.	Zufallswert	1418	
5.2.3.	JScript Laufzeit-Bibliothek (ActiveXObject) des Internet Explorer	1420	
5.2.3.1.	Dictionary Objekt	1427	
5.2.3.2.	FileSystemObject Objekt	1430	
5.2.3.2.1.	FileSystemObject.Drive Objekt	1434	
5.2.3.2.2.	FileSystemObject.Drives Collection	1435	
5.2.3.2.3.	FileSystemObject.File Objekt	1437	
5.2.3.2.4.	FileSystemObject.Folder Objekt	1439	
5.2.3.2.4.1.	FileSystemObject.Folder.Files Collection	1441	
5.2.3.2.4.2.	FileSystemObject.Folder.Folders Collection	1442	
5.2.3.2.5.	FileSystemObject.TextStream Objekt	1442	
5.2.3.2.6.	FileSystemObject und Windows Script Host (WSH)	1445	
5.2.3.2.6.1.	Beispiel: Zugriff auf Recent-Ordner	1445	
5.2.3.2.6.2.	Beispiel: Zugriff auf Registry	1447	
5.2.3.2.6.3.	Beispiel: Ordner erzeugen	1450	
5.2.3.2.6.4.	Beispiel: Lokales Programm starten	1452	
5.2.3.2.6.5.	Beispiel: Tastensimulation	1455	
5.2.3.2.6.6.	Beispiel: Zugriff auf Kontextmenü des Windows Explorers	1457	
5.2.3.2.6.7.	Beispiel: Zugriff auf PATH-Variable	1460	
5.2.4.	Windows Media Player 7.1 und Internet Explorer	1460	
5.2.4.1.	Begriffe	1468	
5.2.4.1.1.	Media Datei	1468	
5.2.4.1.1.1.	Media Datei als nicht window-spezifische Media Datei	1469	
5.2.4.1.1.2.	Media Datei als window-spezifische Media Datei	1469	
5.2.4.1.1.2.1.	Media Datei als Windows Media Datei	1469	
5.2.4.1.1.2.2.	Media Datei als Windows Meta Datei	1469	
5.2.4.1.2.	Media Bibliothek	1470	
5.2.4.1.3.	Playliste	1470	
5.2.4.1.4.	JScript: media Objekt, Media Item Objekt und Collection mediaCollection	1470	
5.2.4.2.	Varianten des Windows Media Player (der ActiveX-Controls)	1470	
5.2.4.3.	Instanziierung des Windows Media Player im HTML-Dokument	1474	
5.2.4.3.1.	Belegung CLASSID-Attributim OBJECT-Tag	1474	
5.2.4.3.2.	ID-Attribut für die Referenz auf die Instanz des Windows Media Players	1474	
5.2.4.3.3.	HTML-Vorbelegung von Eigenschaften der Instanz des Windows Media Players	1474	
5.2.4.3.4.	HTML-Attribute WIDTH und HEIGHT im OBJECT-Tag	1475	
5.2.4.3.5.	automatische Fehleranzeige	1475	
5.2.4.3.6.	Beispiel	1475	
5.2.4.4.	Eventbehandlung	1477	
5.2.4.5.	Objekte, Collectionen und Events des Windows Media Player	1478	
5.2.4.5.1.	Objekt des Windows Media Players (ID_Player)	1478	
5.2.4.5.2.	ID_Player.cdromCollection Collection	1483	
5.2.4.5.3.	ID_Player.closedCaption Objekt	1484	
5.2.4.5.4.	ID_Player.controls Objekt	1484	
5.2.4.5.5.	ID_Player.currentMedia Objekt	1486	
5.2.4.5.6.	ID_Player.currentPlaylist Objekt	1488	
5.2.4.5.7.	ID_Player.error Objekt	1491	
5.2.4.5.8.	ID_Player.mediaCollection Collection	1491	
5.2.4.5.9.	ID_Player.network Objekt	1493	
5.2.4.5.10.	ID_Player.playlistCollection Collection	1494	
5.2.4.5.11.	ID_Player.settings Objekt	1496	
5.3.	Webspeech von Logox im Internet Explorer und Netscape	1498	
5.3.1.	Webspeech-Objekt erzeugen	1500	
5.3.1.1.	Webspeech-Objekt in HTML erzeugen	1506	
5.3.1.1.1.	Parameter AUTHKEY (ab Webspeech 4)	1507	
5.3.1.1.2.	Parameter TEXT und URL	1509	
5.3.1.1.3.	Parameter AUTOSTART	1510	
5.3.1.1.4.	Parameter IMMEDIATE	1510	
5.3.1.1.5.	Parameter MOUTHANIMATION	1510	
5.3.1.1.6.	Parameter MOUTHCOLOR	1510	
5.3.1.1.6.	Parameter TEXTANIMATION	1510	
5.3.1.1.7.	Parameter TEXTCOLOR	1510	
5.3.1.1.8.	Parameter TEXTPOSITION	1511	
5.3.1.1.9.	Parameter TEXTSIZE	1511	



5.3.1.1.10. Parameter BACKGROUND COLOR	1511
5.3.1.1.11. Parameter OPAQUE	1511
5.3.1.1.12. Parameter CONTROLPOSITION	1511
5.3.1.2. Webspeech-Objekt in Javascript erzeugen	1511
5.3.2. Webspeech-Objekt in Javascript verwalten	1511
5.3.2.1. Webspeech-Objekt in Javascript erkennen	1511
5.3.2.2. Webspeech-Objekt in Javascript programmieren	1513
5.3.2.2.1. Webspeech-Objekt und seine Methoden	1513
5.3.2.2.2. Webspeech-Objekt und Events	1519
5.3.2.2.3. Beispiel zur Javascript-Programmierung zu Webspeech 2 für IE und NS 4.x	1519
5.3.3. Webspeech-Sprechts (Auswahl)	1528
5.3.3.1. Webspeech-Sprechts unter Webspeech 4	1528
5.3.3.2. Webspeech-Sprechts in Webspeech 4	1529
5.4. Beispiel für windowseigenes Active-X-Control – Analoge Uhr	1530
6. Anhang: Eigenschaften und Methoden des Internet Explorer	1548
7. Anhang: Styles des Internet Explorer	1915
7.1. Objekte mit STYLE-Attribut	1916
7.2. Style-Eigenschaften - Übersicht	1917
7.3. Style-Methoden	1951
8. Anhang: Events des Internet Explorer	1953
8.1. wichtige Objekte und deren Events(Auswahl) - Übersicht	1953
8.2. wichtige Events und deren Auftreten in Objekten - Übersicht	1964
8.3. Einzelbeschreibungen der Events (teilweise mit Beispielen)	1977
9. Anhang: Filter des Internet Explorer	2001
10. Anhang: Eigenschaften und Methoden des Windows Media Player 7.1	2009
Index	2016



1. Dialekte und Versionen von Javascript

Ein Dialekt entspricht einer Version der Scriptmaschine des jeweiligen Browser-Herstellers.

Herstellerspezifische Versionen von Javascript - Ursachen, Vor- und Nachteile, Risiken für Web-Designer

Die Scriptmaschine ist eine Softwarekomponente (z.T. auch eine Komponente des Betriebssystems), die den Quelltext liest, interpretiert (parst) und in durch den Browser ausführbare Befehle umwandelt (z.B. Anzeige von HTML-Elementen). Die Scriptmaschine erzeugt keinen Maschinencode wie ein Compiler, sondern benutzt vordefinierte Run-Time-Bibliotheken (Objekte).

Die Browser-Version ist immer an die Version der Scriptmaschine gebunden. Implementationen in einer jüngeren Version einer Scriptmaschine müssen vom Browser älteren Datums nicht unbedingt nutzbar sein. Dafür kann die Scriptmaschine zu älteren Browsern abwärtskompatibel sein, muss es aber nicht sein. Browserinterne Elemente (Objekte) hängen also von der Scriptmaschine ab, die diese Objekte implementiert hat. Was die Scriptmaschine nicht kennt, kann der Browser erst recht nicht wissen und können.

Grundsätzlich gilt: Browser-Hersteller implementieren JavaScript immer mit browser-spezifischen Eigenschaften, die bewusst von anderen Browsern abgrenzen sollen, auch wenn ansonsten auf Kompatibilität geschworen wird. Die Abgrenzung nennt sich dann u.a. Browsermodernisierung im Rahmen der "innovativen" Standardpflege per Konsortium aus Browserherstellern. Dass also Standardpflege bewusst vom Standard wegmuiert, ist so möglich, indem z.B. der Standard über zig Jahre nicht mehr weitergepflegt wird und durch inzwischen andere heranschleichende, ev. innovative Hersteller-"Standards" ersetzt wird - XHTML ist ein Beispiel dafür. - Letztendlich geht es nur um Geld aus Rendite und Marktanteilen (wozu auch gern mal Open Source benutzt wird).

Noch cleverer macht es Microsoft: Der Browser ist konzeptionell eine abgeleitete Instanz des Windows Explorers als Instanz des Arbeitsplatzes also der GUI-Shell (Pendant in Linux z.B. KDE)). Microsoft lässt z.B. JScript durch JScript.Net innovativ ersetzen (also auch die Art der Komponenten des Browsers bzw. von Windows und deren Zugriff über JScript) und schaltet zusätzlich Komponenten von Windows wie Active-X-Control, die bisher per älterem JScript aufrufbar waren, per Definition (z.B. wegen Sicherheitsproblemen oder Patentrechtswahrung) z.T. ersatzlos ab, oder implementiert den Zugriff auf neue Komponenten nicht mehr in JScript außerhalb der Net-Umgebung: Programmierer werden dann regelrecht über den Tisch gezogen, wenn ein bisher funktionierendes Script mit Referenz z.B. auf ein Active-X-Control wegen abgeschalteten Bibliotheken des Control nicht mehr funktioniert, oder wenn z.B. eine JScript-Komponente im HTML-DOM unter neuerem Browser Fehler verursacht, obwohl sich der Syntax im HTML-DOM nicht verändert hat (konkrete Beispiele siehe unten). Man beachte zusätzlich, dass Veränderungen und Abschaltungen von vom Windows-Nutzer nicht oder doch eingespielten Patches abhängen können, wobei fehlerhafte Patches und deren eventuelle Bereinigung durch Nachfolgepatch eingeschlossen sind (konkretes Beispiel siehe unten Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen).

Die Schnittmenge der JavaScript-Derivate sollte also die Kompatibilität umfassen, tut es aber nicht.

Zusätzlich erschwerend ist der Umstand, dass namensgleiche JavaScript-eigene Komponenten differenziert implementiert sein können.

Es gibt aber eine Schnittmenge, die zwar unbequem (weil mit Programmierungs- und Pflegeaufwand verbunden) ist, dafür den Scriptcode portierbar hält: Man benutze Standard-Komponenten von Javascript, die allen Browsern bekannt sind. Für Browserspezifische Komponenten benutze man Bibliotheken (JS-Dateien), die je nach Browser ausgewechselt werden und deren Schnittstellen aus Standard-Komponenten bestehen.

Man muss also **dynamisch programmieren**, auch um Kompatibilitätsprobleme lösen zu können.

Analog zu den Standard-Script-Komponenten gibt es HTML-DOM-Funktionen, die von verschiedenen Browsern beherrscht werden - natürlich auch hier z.T. divergent z.B. der Zwang zur Nutzung der .getElementByXXXX-Funktionen in Nicht-Microsoft-Browsern, wobei der MS Internet

Explorer diese Funktionen auch kennt, aber bereits den Wert des ID-Attributes im HTML-Tag als Zeiger im Script zulässt (deswegen hat man ja im HTML-Code das ID-Attribut hinterlegt, zumal diverse HTML-Tags das ID-Attribut besitzen). Also benutze man auch hier JS-Dateien als Bibliotheken je nach Browser-Version (z.B. eine Funktion für ein im Body vorhandenes Objekt, die beim IE eval('var Zeiger='+ID_Attribut_Wert+'); return Zeiger; und bei anderen Browsern return document.body.getElementById(ID_Attribut_Wert); benutzt, wobei ID_Attribut_Wert aus dem Funktionsargument stammt. Würde nur für den IE programmiert werden, würde kein Code anfallen, da ID sofort einsetzbar ist. Nur wegen anderer Browser und Code-Reduzierung ist auch im IE .getElementById() zu verwenden (Tipp: Zeiger in (globale) Variable (z.B. als Zeigerfeld-Element) speichern, oder grundsätzlich createElement() verwenden und den dadurch gelieferten Zeiger speichern müssend, wobei dann ein ID-Attribut im HTML-Code des Argumentes von createElement() keinesfalls kodiert sein darf. (doppelte Zeiger unzulässig)).

Beispiel: Netscape und Internet Explorer

Javascript-Versionen und -Dialekte zeigen z.T. sehr deutlich die unlösbaren Divergenzen zwischen den Browsern von Netscape und Microsoft. Beide Hersteller verändern z.T. inkompatibel ihre Script-Versionen gegenüber dem Javascript-Standard.

Es ist möglich, für beide Browser per Javascript-Standard zu programmieren. Dabei muss verzichtet werden

bezüglich des Internet Explorer ab 5.5	auf Erweiterungen zum Browser
bezüglich des Netscape ab 6.x	auf die Java-Klassen-Einbindung (z.B. von Sun und Netscape) für Plugins

Es ist unmöglich, ein und denselben Javascriptcode gleichzeitig für beide Browser **ohne** eine Browserunterscheidung zu verwenden. Es muss vor allem dann doppelt programmiert werden, wenn nicht der Javascript-Standard verwendet werden soll.

Näheres zu Javascript, Skriptmaschine und Objekten ist in den Beschreibungen zu den Objekten in Javascript/JScript und des Browsers zu finden. Dort werden auch ein Ansatz zur objektorientierten Programmierung mit Javascript geliefert und die Ursachen für die Existenz von Javascript-Dialekten bzw. -Versionen erklärt.



SCRIPT-Tags sind **nicht** verschachtelbar (auch nicht innerhalb eines Javascript-Codes). Man kann also **nicht** mit Javascript einen **neuen** Script-Block **innerhalb eines vorhandenen** Scriptblockes erzeugen, wenn dieser Block, der den Javascriptcode zur Script-Blockerzeugung enthält, gerade geparkt wird (auch eval() funktioniert nicht). Dabei ist es egal, ob der Javascript-Code im HEAD oder BODY liegt. Sollte ein Browser doch auf o.g. Art einen inneren Scriptblock erzeugen können, so ist der Browser in diesem Fall nicht kompatibel. Außerdem kann es sein, dass ein Script-Tag, der innerhalb eines Javascriptcodes (z.B. innerhalb von document.write()) oder nach dem Kommentarzeichen // kodiert wurde, vom Browser als HTML-Tag erkannt und als solches eigenständig und unabhängig vom Kodierungskontext ausgeführt wird.

Für Programmierer unter Windows XP mit dem Internet Explorer bitte **unbedingt** beachten: Die Scriptmaschine unter Windows XP ist wesentlich **weniger fehlertolerant** als die Scriptmaschine unter Windows 98 bei identischer Browserversion und identischem Patch-Stand der Browsersoftware. Unter Windows XP ist der Internet Explorer 6.x implementiert. JScript-Code, der unter Windows 98 einwandfrei funktioniert, muss es unter Windows XP **nicht** ! Javascript-Code, der unter Windows XP funktioniert, wird es auch unter Windows 98 tun. Mit anderen Worten: Die Scriptmaschine unter Windows XP ist bezüglich Fehler **nicht** abwärtskompatibel ! Deswegen bitte **unbedingt** den Scriptcode unter Windows XP testen !

Beispiel: Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler



Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()
 Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.
 Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.
 Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

- Scriptfehleranzeige ist erlaubt im IE 7
- Popupblocker ist im IE abgeschaltet
- ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.
- ein weiteres Fenster (Register) z.B. leere Seite (about:blank)
- beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt, bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren oder Popsups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten
 weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern eingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus
 onblur
 onfocusin
 onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.



Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X-Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)



Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen: • 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen: • 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen: • http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp (http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Umeine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen: •

<http://www.microsoft.com/technet/security/bulletin/ms99-037.mspx>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.mspx>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.mspx>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.mspx>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.



Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5



MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87=parent.Y_unload(0); X86[0]=new Function("X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet, so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild

<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12)">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12)">

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.



Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein, innerhalb dessen dann die neuen HTML-Elemente erzeugt werden.

Hinweis zum Einrichten eines virtuellen Hosts per Apache-HTTP-Server:

Anstelle des HTTP-Webservers vom Internetprovider kann der eigene PC als Test-Server fungieren, um z.B. die Webseite so zu testen, als wäre sie gerade online auf dem Webserver.

Der Hobbyprogrammierer will u.a. kostengünstig testen, also bieten sich die kostenlosen HTTP-Server an, die einen virtuellen Host anhand eines beliebigen Ordners auf der Festplatte erzeugen können, wobei Local Host (127.0.0.1) als virtueller Host einrichtbar sein muss (der nicht von der Firewall-Software des PC allein verwaltet sein darf) und die zu testende Webseite im Browser per Domainnamen aktivierbar sein muss (anstelle der Eingabe von 127.0.0.1). Man google, um festzustellen, welche Produkte diesen Kriterien entsprechen. - Abkürzend fällt die Wahl nicht zufällig auf den HTTP-Server von Apache (www.apache.org), welcher ziemlich schwierig zu konfigurieren ist, wenn mehr als nur localhost genutzt werden soll (Für Hobbyzwecke reicht localhost aus).

Der Hobbyprogrammierer, welcher für den Microsoft Internet Explorer (ab IE 7 heißt der Windows Internet Explorer) in seinen Varianten je nach Windows-Version programmieren will, wird definitiv folgende Probleme bekommen:

Microsoft lässt u.a. die Installation des Internet Explorers in parallelen Versionen nicht zu, obwohl Browserversionen nachweislich nicht kompatibel sind und Microsoft Browserversionen innerhalb Windowsversionen supportet werden. Daher benötigt man pro Version des Internet Explorers eine Windows-Installation. Pro Windows-Installation wird eine Windows-Lizenz fällig, auch wenn auf anderer Festplatte am ansonsten identischen PC installiert wird (Windows-Online-Update erkennt solche Doppelversionen und verweigert den Support). Auch wer Windows unter VM emulieren will (oder auf Apple mit Intel-Technik Windows parallel mit Apple nutzen will), benötigt eine Lizenz. Mit anderen Worten: Auch wenn die Mehrfachinstallation nicht parallel nutzbar wäre, sondern immer nur genau 1, will Microsoft Geld haben - nicht umsonst ist Microsoft-Chef so beliebt wie reich und nicht umsonst migrieren immer mehr IT-Anwender zu Linux-Derivaten, die fast identische Browser bezüglich Windows haben. Ergo, der Hobbyprogrammierer wird wohl sämtliche Bekannte mit Testwünschen nerven, oder illegal testen, oder auf andere Browser-Hersteller und deren HTML- sowie Script-Versionen ausweichen, die nicht nur Parallelinstallationen des Browsers zulassen (z.B. Opera unter Windows), sondern auch noch ziemlich gut kompatibel sind (weil identische Scriptmaschine nutzend). Dass Microsoft eventuell keinen kostenlosen HTTP-Server anbietet (der ansonsten auch noch Javascript, Active-X-Control- Aufrufe per Script, DirectX-Zugriff kennen müsste), ist das kleinere - vor allem lösbarere Problem: Eben ohne Microsoft.

Pfade für Dateien im Script einer Webseite, die per HTTP-Server oder lokal von Festplatte gestartet wird:

Relative Pfade per '..' sind unabhängig davon, wie die Wurzel (Root) der Webseite heißt.
Absolute Pfade sind abhängig davon, wie die Wurzel (Root) der Webseite heißt.

Wird die Webseite (z.B. www.twseite.de) von einem HTTP-Server gestartet, dann sind absolute Pfade bezüglich http://www.twseite.de möglich, wobei http://www.twseite.de die Root der Webseite darstellt.

(Achtung: Auf einem HTTP-Server des Internetproviders liegt die Webseite natürlich auch auf einer Festplatte, also dort in einem Ordner. Dieser Ordner muss also mit www.twseite.de logisch verbunden sein - wie, das teilt der Internet-Provider mit.

Tipp: Namen des Ordners auf der Festplatte des Internet-Providers kann man genauso nennen wie den des Ordners auf der lokalen Festplatte des PC. Ordnernamen mit Pfadzeichen wie ':' oder '/' sind natürlich nicht erlaubt)

Wird die Webseite (z.B. www.twseite.de) von lokaler Festplatte aus gestartet, dann sind absolute Pfade bezüglich http://www.twseite.de nicht möglich, da es solchen Ordnernamen nicht geben kann.
Festplattenordner möglich z.B. c:\twseite\

Will man die Webseite identisch verwalten, egal ob man die Webseite von einem HTTP-Server oder von lokaler Festplatte aus startet, dann schaue man sich folgendes Beispiel für www.twseite.de an, das allerdings mit JavaScript oder JScript realisiert wird. Der Festplattenordner ist c:\twseite.

```
var BrowserAufOnlinePruefen=true;           // false für Browser nicht auf online prüfen
                                              // online: Webseite wurde auf HTTP-Server aktiviert
var DomainOhneHTTP='www.twseite.de';        // Host der Webseite ohne http:// und ohne Port
var BrowserIstOnline=false;                 // Annahme: Browser ist nicht online
var PfadDerDateien="";                      // Annahme: Browser ist nicht online also
                                              // alle Pfade unterhalb von c:\twseite\
                                              // wobei die Startdatei index.html der
                                              // Webseite eben in diesem Ordner liegt

if(BrowserAufOnlinePruefen)                 // wenn auf online geprüft werden soll
```



```

{if(window.location.hostname!=null);    // aktuell gefundener Host
  BrowserIstOnline=(window.location.hostname== DomainOhneHTTP);}
                                     // Host www.xxx.yyy prüfen auf aktuelle gefundenen Host
                                     // true so Browser online
}

if(BrowserIstOnline) {PfadDerDateien='http://' + DomainOhneHTTP;}
                                     // Pfad wenn Browser online ist: Alle Pfade unterhalb von
                                     // http://www.twseite.de
                                     // Für den HTTP-Server liegt die Webseite natürlich
                                     // auf einer Festplatte, also dort in einem Ordner.
                                     // Jeder Pfad in der Webseite wird in den Ordnerpfad
                                     // der Festplatte automatisch konvertiert (HTTP zu
                                     // Festplatte per Dienst: Daher der Name HTTP-
                                     // Server)

```

Nachfolgend die beispielhafte Einrichtung einer Webseite per Apache-HTTP-Server 2.2.2 bis 2.2.4. unter Windows XP ab SP 1:

Webseitendomain heißt www.twseite.de mit index.html als Startdatei
 Festplattenordner der Domain, die als virtueller Host über localhost (also 127.0.0.1) laufen soll
 c:\twseite\
 Apache wurde installiert unter e:\wxp\apache\
 wobei gelten muss
 DNS-Dienst von Windows muss aktiv sein (in der Regel ist der Dienst automatisch aktiv)
 Installationstyp All Users, on Port 80, as a Service - Recommended
 Domain und Servername 127.0.0.1 (nicht localhost kodieren)
 Email beliebig@localhost
 Mit der Installation wird der Dienst-Monitor von Apache bei Windowsstart ebenfalls starten.
 Es wird Apache als permanenter Dienst eingerichtet, der über den Dienst-Monitor von Apache
 aktivierbar / deaktivierbar ist. Es sind zwar mehrere Apache-Dienste einrichtbar, aber genau 1
 kann nur immer aktiv sein.

Apache-Software einrichten:

Dienst-Monitor von Apache: siehe oben

aber: Da in Windows für jeden Aufruf der Webseite www.twseite.de unter Apache eine
 Einstellung getroffen werden muss, empfiehlt es sich, den Apache-Dienst
 grundsätzlich manuell zu starten und zu stoppen anhand nachfolgend
 vorgestellter BAT-Dateien.

Dafür muss aber einmalig folgendes eingestellt werden:

Der Apache-Dienst-Name ist per Dienst-Monitor zu sehen (und der Status ob
 aktiv oder deaktiv).

Dienste sind unter Windows per Systemsteuerung-Verwaltung-Dienste
 verwaltbar (auch per Apache-Monitor ist die Dienstverwaltung
 aktivierbar).

Der Apache-Dienst muss zuerst deaktiviert werden, dann auf Starttyp manuell
 gesetzt werden: Mit Windows-Start startet der Dienst nicht automatisch.

Windows anpassen an den Virtuellen Host

Die Anpassung erfolgt so, dass Apache per Batch-File, die unten erklärt werden, gestartet
 und deaktiviert wird (Batch-Files sind passend zur Anpassung von Windows).

Unter c:\windows\system32\drivers\etc\ liegt die Datei hosts

Copyright (c) 1993-1999 Microsoft Corp.

```

#
# Dies ist eine HOSTS-Beispieldatei, die von Microsoft TCP/IP
# für Windows 2000 verwendet wird.
#
# Diese Datei enthält die Zuordnungen der IP-Adressen zu Hostnamen.
# Jeder Eintrag muss in einer eigenen Zeile stehen. Die IP-
# Adresse sollte in der ersten Spalte gefolgt vom zugehörigen
# Hostnamen stehen.
# Die IP-Adresse und der Hostname müssen durch mindestens ein
# Leerzeichen getrennt sein.
#
# Zusätzliche Kommentare (so wie in dieser Datei) können in
# einzelnen Zeilen oder hinter dem Computernamen eingefügt werden,

```



```
# aber müssen mit dem Zeichen '#' eingegeben werden.
#
# Zum Beispiel:
#
#      102.54.94.97      rhino.acme.com      # Quellserver
#      38.25.63.10      x.acme.com          # x-Clienthost
127.0.0.1      localhost
```

Diese Datei verwaltet -wie man sieht - auch den localhost.

Standardgemäß ist localhost auf 127.0.0.1 gelegt (nur deswegen sind localhost und 127.0.0.1 synonym)

Soll aber die Webseite www.twseite.de über 127.0.0.1 getestet werden, muss also die Zeile

127.0.0.1 localhost

ersetzt werden durch

127.0.0.1 www.twseite.de

Nach dem Test muss localhost wieder für 127.0.0.1 verfügbar gemacht werden, also der Standard gesetzt werden.

Genau dieses Ersetzen machen die Batch-Files siehe unten. Dafür benötigen sie nur 2 neue Ordner, die einmalig mit Inhalt manuell angelegt werden müssen in c:\windows\system32\drivers\etc\

Schritt 1: Ordner c:\windows\system32\drivers\etc_hosts_standard\

manuell erzeugen, und dorthin die bisher unveränderte, also originale hosts-Datei kopieren (Datei enthält 127.0.0.1 localhost)

```
# Die IP-Adresse und der Hostname müssen durch mindestens ein
# Leerzeichen getrennt sein.
#
# Zusätzliche Kommentare (so wie in dieser Datei) können in
# einzelnen Zeilen oder hinter dem Computernamen eingefügt werden,
# aber müssen mit dem Zeichen '#' eingegeben werden.
#
# Zum Beispiel:
#
#      102.54.94.97      rhino.acme.com      # Quellserver
#      38.25.63.10      x.acme.com          # x-Clienthost
127.0.0.1      localhost
```

Schritt 2 Ordner c:\windows\system32\drivers\etc_hosts_mit_www_twseite_de\ manuell erzeugen, und dorthin die bisher unveränderte, also originale hosts-Datei kopieren

(Datei enthält 127.0.0.1 localhost)

dann diese Datei in diesem Ordner per notepad.exe (nicht

Word etc.) auf

127.0.0.1 www.twseite.de

```
# Copyright (c) 1993-1999 Microsoft Corp.
#
# Dies ist eine HOSTS-Beispieldatei, die von Microsoft TCP/IP
# für Windows 2000 verwendet wird.
#
# Diese Datei enthält die Zuordnungen der IP-Adressen zu Hostnamen.
# Jeder Eintrag muss in einer eigenen Zeile stehen. Die IP-
# Adresse sollte in der ersten Spalte gefolgt vom zugehörigen
# Hostnamen stehen.
# Die IP-Adresse und der Hostname müssen durch mindestens ein
# Leerzeichen getrennt sein.
#
```



Zusätzliche Kommentare (so wie in dieser Datei) können in
 # einzelnen Zeilen oder hinter dem Computernamen eingefügt werden,
 # aber müssen mit dem Zeichen '#' eingegeben werden.

 # Zum Beispiel:

```
#      102.54.94.97      rhino.acme.com      # Quellserver
#      38.25.63.10      x.acme.com          # x-Clienthost
```

```
127.0.0.1      www.twseite.de
```

Virtual-Host genau 1x einrichten in e:\wpx\apache\conf\ dort in der Datei httpd.conf

erst httpd.conf kopieren in einen Sicherungsordner freier Wahl

dann per notepad.exe (nicht per Word etc.) am Ende der httpd.conf folgenden Text einfügen

Achtung: Es muss natürlich dann
 www.twseite.de ersetzt werden durch zu
 testende Domain
 c:/twseite ersetzt werden durch den
 wirkliche Pfad

Virtual Host für www.twseite.de auf 127.0.0.1:80 gehostet

ERST Virtual Host für den Server, der damit alle Servernamen ungleich
 www.twseite.de abfängt

also Angaben aus Serverinstalltion verwendet

NameVirtualHost 127.0.0.1

```
<VirtualHost 127.0.0.1>
  ServerName localhost
  ServerAlias 127.0.0.1
  DocumentRoot e:/wpx/apache/htdocs
  ErrorLog e:/wpx/apache/logs/error.log
  TransferLog e:/wpx/apache/logs/access.log
  ScriptAlias /cgi-bin/ e:/wpx/apache/cgi-bin/
</VirtualHost>
```

Virtual Host für www.twseite.de

```
<VirtualHost 127.0.0.1>
  ServerName www.twseite.de
  ServerAlias 127.0.0.1
  DocumentRoot c:/twseite
  DirectoryIndex index.html
  ErrorLog c:/twseite/apache_error.log
  TransferLog c:/twseite/apache_access.log
  <Directory c:/twseite>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
  </Directory>
</VirtualHost>
```

Apache-Start und Stop per folgender BAT-Dateien, die z.B. in C:\ liegen (egal wo auf Festplatte):

Achtung: Falls der Start von Apache wegen nicht vorhandenem Dienst nicht erfolgen kann, dann gilt:

Mit der Apache-Installation wurde ein Dienst eingerichtet: siehe oben. Nur leider, der Apache-Start verlangt einen weiteren Dienst mit anderem Namen. Da bekannt ist, dass aber nur 1 Dienst zu jedem Zeitpunkt aktiv sein kann, ist es sehr verwunderlich, wieso Apache-Start den bereits vorhandenen Dienst nicht nimmt.

Anstelle von wundern bitte folgendes genau 1 mal ausführen:



```
e:\wyp\apache\bin\ httpd.exe -k install
```

im bin-Verzeichnis liegen die ausführbaren Dateien von Apache.
httpd.exe ist die Hauptkomponente von Apache.
-k install installiert einen weiteren Apache-Dienst.

Logischerweise ist dieser Dienst analog zum Dienst, der bei Apache-Installation erzeugt wurde,
ebenfalls per Dienste-Verwaltung zu deaktivieren und auf Starttyp manuell zu setzen.
Dieser neue Dienst hat aber den passenden Namen, der vom Start von Apache akzeptiert wird.

Hier ein Beispiel für den Dienste-Wirrwarr:

Apache-Installation erzeugt Dienst 'Apache2.2'.
Apache-Start will aber den Dienst 'Apache 2'

Batch-File 'ApacheStarten.bat' für Start des Apache, wobei der Dienst 'Apache2' aktiviert wird

```
@echo off
cls
echo Apache als Anwendung starten bei inaktivem Apache-Dienst 'Apache2'
echo          (Apache-Dienst darf nicht bei Windows-Start aktiv sein)
echo          (Apache stoppen immer per ApacheStoppen.bat)
echo hosts-Datei mit www.twseite.de bereitstellen
echo.
copy /V /Y C:\WINDOWS\system32\drivers\etc\_hosts_mit_www_twseite_de\hosts
C:\WINDOWS\system32\drivers\etc\hosts > NUL
e:\wyp\apache\bin\httpd.exe -k start
echo.
echo.
echo      ..... Apache-Anwendung wurde nur gestartet, wenn oben keine
echo              Fehlermeldung angezeigt wurde !
echo              Falls Fehlermeldung dann im Monitor Apache Servers
echo              zu allen Eintraege STOP-Button klicken (falls klickbar)
echo              und Batch-File neu starten
echo.
echo.
echo      Apache kann keine ActiveX durchreichen (z.B. Logox WebSpeech)
echo              per JScript-erzeugte Popups nicht immer korrekt
rendern
echo.
echo.
pause
echo.
echo.
echo aktuelle hosts-Datei lautet
echo.
type C:\WINDOWS\system32\drivers\etc\hosts
echo.
pause
```

Batch-File 'ApacheStoppen.bat' für Start des Apache, wobei der Dienst 'Apache2' de-aktiviert wird

```
@echo off
cls
echo Apache als Anwendung stoppen bei aktiven Apache-Dienst 'Apache2'
echo          (Apache-Dienst aktiviert per ApacheStarten.bat)
echo hosts-Datei ohne www.twseite.de bereitstellen
echo.
copy /V /Y C:\WINDOWS\system32\drivers\etc\_hosts_standard\hosts
C:\WINDOWS\system32\drivers\etc\hosts >NUL
e:\wyp\apache\bin\httpd.exe -k stop
echo.
echo.
echo      ..... Apache-Anwendung wurde nur gestoppt, wenn oben keine
```



```
echo                Fehlermeldung angezeigt wurde !
echo                Falls Fehlermeldung dann im Monitor Apache Servers
echo                dann war kein Appache-Server aktiv.
echo.
echo.
pause
echo.
echo.
echo.
echo aktuelle hosts-Datei lautet
echo.
type C:\WINDOWS\system32\drivers\etc\hosts
echo.
pause
```

Batch-File 'ApacheHostsDateiAnzeigen.bat' für Anzeige der hosts-Datei

```
@echo off
cls
echo aktuelle hosts-Datei anzeigen
echo.
type C:\WINDOWS\system32\drivers\etc\hosts
echo.
echo.
echo.
pause
```

Webseite aktivieren per Apache:

Da die Webseite nur per o.g. Batch-Dateien gestartet bzw. gestoppt werden kann, kann der Apache-Dienst-Monitor nur noch zur Kontrolle benutzt werden, ob eine Apache-Dienst aktiv ist.

Schritt 1: Vor dem Start der Webseite darf kein Apache-Dienst aktiv sein

Schritt 2 ApacheHostsDateiAnzeigen.bat aktivieren. Die hosts-Datei muss enthalten

127.0.0.1 localhost

Schritt 3: ApacheStarten.bat aktivieren. Apache startet.

Wer will kann jetzt Schritt 2 wiederholen und sieht dann

127.0.01. www.twseite.de

Wenn jetzt die Firewall-Software sich meldet, dann Apache erlauben
am Port 80 (also HTTP) von 127.0.0.1

Achtung: Nutzt die zu testende Webseite andere Ports z.B. die von
Plugins, dann können die Plugins selbst einen
Firewalleintrag verlangen, aber Apache kann natürlich
nur HTTP (Port 80) und muss nur zwischen Webseite
und den Plugins vermitteln (eben über Port 80)

Schritt 4: Browser der Wahl aktivieren
Dort die Domain www.twseite.de eintippen die Webseite öffnet sich anhand
c:\twseite\index.html

Tipp: Sollte die Webseite bereits auf einem Webserver online sein, dann würde
ohne Apache-Start natürlich die online-Variante aktiviert werden
und nicht die Webseite von der Festplatte aus dem Ordner c:\twseite !

Wichtig: Sollte die Webseite im Ordner c:\twseite\ geändert worden sein, so
muss der Browser-Cache gelöscht werden, bevor die die Webseite
erneut aktiviert wird. Apache lässt den Browser genauso agieren, als ob
der online die Webseite laden würde, also über den Browser-Cache.
Manche Browser oder Webseiten unterbinden dann das Neuladen einer
Datei, die im Browser-Cache dem Namen nach vorhanden ist, aber in
c:\twseite verändert vorliegt also zwingend neu geladen werden muss....
Ergo Browser-Cache vorher löschen.



Browser-Cache im löschen im
 Internet Explorer per Internetoptionen, die
 im Browser per Menüpunkt Extra oder per Systemsteuerung
 aktivierbar sind (ev. Verknüpfung auf Desktop manuell
 erzeugen)
 Opera per Menüpunkt Extra-Einstellungen-Erweitert

Schritt 5: Webseite getestet (man hat jetzt ein Smily-Gesicht oder graue Haare mehr, oder den Chef,
 Kunden, Frau am Hals *g)
 also Zeit zum Deaktivieren von Apache per ApacheStoppen.bat

Schritt 6: Webseitendaten aus c:\twseite sichern.

1.1. Javascript-Versionen beim Netscape

Die Version der Scriptmaschine wird durch HTML-Angabe des unterstützten Javascript-Standards bekannt gegeben.

<u>HTML-Kodierung der Scriptversion</u>	<u>Javascript-Standard</u>	<u>NS-Version</u>
<SCRIPT LANGUAGE="JavaScript1.0">	Version 1.0	ab NS 2.x
<SCRIPT LANGUAGE="JavaScript1.1">	Version 1.1	ab NS 3.x
<SCRIPT LANGUAGE="JavaScript1.2">	Version 1.2	NS 4.0 bis 4.05
<SCRIPT LANGUAGE="JavaScript1.3">	Version 1.3	NS 4.06 bis 4.70
<SCRIPT LANGUAGE="JavaScript1.4">	Version 1.4	NS 4.7x
<SCRIPT LANGUAGE="JavaScript1.5">	Version 1.5	ab NS 6.x

Wird nur <SCRIPT> kodiert wurde, so gilt
 JScript wenn das HTML-Dokument unter dem IE läuft
 Javascript in der Version laut Browser, wenn das HTML-Dokument unter dem NS läuft

Javascript wird gegenüber dem Javascript-Standard z.B. unter NS ab 6.x browserspezifisch erweitert .

Beispiele: Java-Klassen-Einbindung für Plugins
 Verwaltung signierter Scripts

Netscape kann Erweiterungen des Browsers durch Plugins realisieren, die
 auch über Javascript programmierbar sein **können**, aber nicht müssen
 bezüglich dem IE 6.x **inkompatibel** sind, da dieser keine Plugins mehr unterstützt.

1.2. Javascript-Versionen von Microsoft

Die Version der Scriptmaschine wird durch **keine** HTML-Angabe bekannt gegeben.

JScript kennt auch keine HTML-Angaben für die Sprachversionen von JScript. Es muss **nur** kodiert werden:

```
<SCRIPT LANGUAGE="JScript">
```

Wird nur <SCRIPT> kodiert wurde, so gilt
 JScript wenn das HTML-Dokument unter dem IE läuft
 Javascript in der Version laut Browser, wenn das HTML-Dokument unter dem NS
 läuft

Hinweis: Wenn unter dem IE <SCRIPT LANGUAGE="JavaScript1.x"> (x ist zu ersetzen mit der
 Versionsnummer) **und** zugleich JScript-spezifische Elemente kodiert wurden, dann erfolgt Verwendung von JScript.

JScript ist nur z.T. kompatibel zum Javascript-Standard:

<u>Javascript-Standard</u>	<u>IE-Version</u>
JavaScript 1.0	ab IE 3.0
JavaScript 1.1	ab IE 3.02 teilweise
JavaScript 1.2	ab IE 4.0
JavaScript 1.3	ab IE 5.0
JavaScript 1.4	ab IE 6.0
JavaScript 1.5	unklar ab wann und ob überhaupt noch

JScript erweitert browserspezifisch den Sprachumfang gegenüber dem Javascript-Standard
 integriert optional Teile des Betriebssystems und Addons von Windows z.B.

Windows Media Player
 Direct Animation
 Active-X-Controls (ab IE 6.x als Totalersatz für Plugins)

ist eine Schnittstelle zur Erweiterung des Internet Explorers, der in Windows z.T. integriert ist.
 unterstützt ab dem IE 6.x Plugins als Browsererweiterungen **nicht** mehr



Für Programmierer unter Windows XP mit dem Internet Explorer bitte **unbedingt** beachten: Die Scriptmaschine unter Windows XP ist wesentlich **weniger fehlertolerant** als die Scriptmaschine unter Windows 98 bei identischer Browserversion und identischem Patch-Stand der Browsersoftware. Unter Windows XP ist der Internet Explorer 6.x implementiert. JScript-Code, der unter Windows 98 einwandfrei funktioniert, muss es unter Windows XP **nicht** ! Javascript-Code, der unter Windows XP funktioniert, wird es auch unter Windows 98 tun. Mit anderen Worten: Die Scriptmaschine unter Windows XP ist bezüglich Fehler **nicht** abwärtskompatibel ! Deswegen bitte **unbedingt** den Scriptcode unter Windows XP testen !

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read"-Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show()erzeugt.



ein weiteres Fenster (Register) z.B. leere Seite (about:blank)
 beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt, bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren oder Popsups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

- Popupblocker einschalten
- weitere Informationen
- jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

- onfocus
- onblur
- onfocusin
- onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')      // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
{document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.
 Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.



Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:
 Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes



- http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)
http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.mspx>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.mspx>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.mspx>
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement



abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht



mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}
```

```
var X85=new Array();var X86=new Array();
```

```
X85[0]=window.open(...);
```

```
var X87='parent.Y_unload(0);'; X86[0]=new Function(",X87);
```

```
X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet, so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">  
unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild
```

```
<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12) ">  
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12) ">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,

innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

1.3. Feststellung des Dialektes von Javascript

1.3.1. Schritt 1: Erkennung des Browserherstellers

1.3.1.1. Browsererkennung anhand des Browsernamen (navigator.appName)

Diese Art der Browsererkennung ist nicht zuverlässig.



Es wird das Objekt navigator.appName benutzt, dessen Implementation sich aber bezüglich der Werte ändern kann (je nach Version der Scriptmaschine).

Beispiel:

```
<SCRIPT LANGUAGE="JavaScript">
<!--      var ie4=0;                                /* Annahme: Netscape gefunden */
            if (    (navigator.appName.indexOf("Explorer")>=0)    /* auf Explorer */
                &&    (navigator.appVersion.indexOf('4.0')    >=0)    /* der Version 4.0 prüfen */
            )
                ie4=1;
// -->
</SCRIPT>
```

für Netscape bitte "Netscape" verwenden und entsprechende Version

1.3.1.2. **Browsererkennung anhand der Unterscheidung browserinterner Objekte**

Diese Art der Browsererkennung ist zuverlässig und verwendet nicht die Browserunterscheidung anhand der Browsernamen.

Internet Explorer und Netscape haben z.T. divergierende Objekte, die vom jeweiligen Konkurrenzprodukt nicht erkannt werden. Das erleichtert eine Browsererkennung per Javascript/JScript.

Andererseits ist im Falle ab Netscape 6.x eine Änderung gegenüber dessen Vorgängern bis 4.7x eingetreten (Netscape-Versionen 5.x existieren nicht). Es wurde mit z.T. Fähigkeiten des verbreiteten Netscape 4.7x konsequent gebrochen, was so mancher Internet-User nicht weiß, der zu dem eventuell noch hartnäckig auf "Netscape schwört". Netscape ab 6.x kann endlich Objekte abbilden, die der Internet Explorer ebenfalls kennt. Basis beider Browser ist die genormte Implementierung von HTML und Script anhand des "Dokument Objekt Modells" (DOM).

Beispiel:

```
// interne Variablen initialisieren
var Browser_Version_Haupt = 0;
var Browser_Version_Unter = 0;

// Informationen zum Useragent holen
//      Objekt navigator.userAgent kennen IE und NS
var UserAgent = navigator.userAgent;

// prüfen ob UserAgent keine Leerkette ist
if (UserAgent != "")
{
    // UserAgent ist keine Leerkette

    // Hauptversion des Browsers holen
    for (var i=0; i < 10; i++)        // Annahme: Eine Hauptversion höher 9 existiert noch nicht
    {
        eval(        'if (UserAgent.indexOf("'      // Version steht vor dem Punkt
                    + i                                // i stellt die zu suchende Hauptversion dar
                    + ',"') != -1)'                // gesucht wird Ziffernfolge mit nachfolgendem Punkt
                                                // i wird automatisch in Ziffernfolge umgewandelt
                    + '{Browser_Version_Haupt = '    // wenn gefunden, so numerischen Wert von i
                                                //      der Variablen zuweisen
                    + i
                    + '};}'
        );
    }

    // Unterversion des Browsers holen
    for (var i=0; i < 100; i++)        // Annahme: Eine Unterversion höher 99 kann nicht existieren
    {
        eval(        'if (UserAgent.indexOf("'      // Unterversion steht nach dem Punkt
                    + i                                // i stellt die zu suchende Unterversion dar
                    + ',"') != -1)'                // gesucht wird Ziffernfolge mit vorausgehendem Punkt
                                                // i wird automatisch in Ziffernfolge umgewandelt
                    + '{Browser_Version_Unter = '    // wenn gefunden, so numerischen Wert von i
                                                //      der Variablen zuweisen
                    + i
                    + '};}'
        );
    }

    // wenn Unterversion des Browsers am Ende eine Null hat, so diese eliminieren z.B. 50 zu 5
    if ((Browser_Version_Unter % 10) == 0)
```



```

    {Browser_Version_Unter = Browser_Version_Unter / 10;} // Bsp.: 50 % 10 = 5 Rest 0, also 50 / 10 = 5
}

// NS selektieren

var NSunter6= ( (!document.all) // kein NS kennt document.all
               || (document.layers) // document.layers nur bis NS unter 6.x
             );

var NSab6= ( (!document.all) // kein NS kennt document.all
            && (document.getElementById) // ab NS 6.x ist getElementById implementiert

            // theoretisch wäre zusätzlich (!document.layers) kodierbar,
            // aber mit Implementation von getElementById wurde gleichzeitig
            // document.layers abgeschafft
          );

var NS = (NSunter6 || NSab6);

var NS4x = ( (NSunter6)
            && (Browser_Version_Haupt >= 4)
            && (Browser_Version_Haupt < 5)
          );

var NS6x = NSab6;

// theoretisch kann noch auf die Hauptversionsnummer geprüft werden
// var NS6x = ( (NSab6)
//             && (Browser_Version_Haupt >= 6)
//             );

// IE selektieren
var IE = (document.all) ? true : false; // if-Anweisung ist wichtig, sonst erfolgt Zeigerzuweisung
// Objekt document.all ist in allen IE-Versionen verfügbar

var IEab5 = ( (IE)
              && (document.getElementById) // ab IE 5.x ist getElementById implementiert
            );

var IE4x = ( (IE) // NS-Browser setzt IE auf false
             && (!IEab5) // damit setzt NS-Browser IEab5 auf false,
                       // also !IEab5 auf true
                       // Es muss (IE) && (!IEab5) abgefragt werden !
                       // Würde nur (!IEab5) kodiert sein,
                       // dann würde der NS-Browser
                       // IE4x auf true setzen.
             && (Browser_Version_Haupt >= 4)
             && (Browser_Version_Haupt < 5)
          );

var IE5x = ( (IEab5)
             && (Browser_Version_Haupt >= 5)
             && (Browser_Version_Haupt < 6)
          );

var IE55 = ( (IEab5)
             && (Browser_Version_Haupt == 5)
             && (Browser_Version_Unter == 5)
          );

var IE6x = ( (IEab5)
             && (Browser_Version_Haupt >= 5)
          );

```

1.3.1.3. Browsererkennung beim Internet Explorer ab IE 5.x

Es werden Tags benutzt, die nur der IE ab 5.x erkennt.

Beispiel 1:

```

<!-- [if IE 5] -->
    javascript_oder_html_code_fuer_den_IE5
<!-- [endif] -->

```




```
<!-- [if ! IE 5] /-->
    javascript_oder_html_code_fuer_andere_Browser    // auch alle IE-Versionen ungleich 5
<!-- [endif] /-->
```

Beispiel 2:

```
<!-- [if IE] /-->
    javascript_oder_html_code_fuer_den_IE
<!-- [endif] /-->

<!-- [if ! IE] /-->
    javascript_oder_html_code_fuer_andere_Browser    // aber keine IE-Versionen
<!-- [endif] /-->
```

Scriptcode muss innerhalb von <SCRIPT ...> ... </SCRIPT> kodiert werden.

HTML-Code kann nur kodiert werden, wenn obige Prüfung innerhalb von BODY liegt.

1.3.2. Schritt 2: Erkennung der Javascript-Version (browserhersteller-spezifisch)

Javascriptversionen -Versionen sind abwärtskompatibel.

JScript-Versionen sind abwärtskompatibel.

Javascript und JScript sind nur z.T. kompatibel.

Bestimmte Features des Browsers lassen sich nur mit bestimmter Script-Version bzw. beim IE manchmal nur mit bestimmter Scriptmaschinen-Version realisieren.

JavaScript 1.0	ab NS 2.x	ab IE 3.0
JavaScript 1.1	ab NS 3.x	ab IE 3.02 teilweise
JavaScript 1.2	NS 4.0 bis 4.05	ab IE 4.0
JavaScript 1.3	NS 4.06 bis 4.70	ab IE 5.0
JavaScript 1.4	NS 4.7x	ab IE 6.0
JavaScript 1.5	ab NS 6.x	steht noch aus

Beispiel:

Hinweise: Bitte keine var-Kodierung von Variablen, da in älteren Versionen eine var-Anweisung unbekannt ist.

Achtung: Kodierung ohne var bewirkt, dass die Variable global ist !

Für jede Scriptversionsprüfung muss ein eigenes <SCRIPT LANGUAGE ...></SCRIPT> kodiert werden.

```
<HEAD>
<SCRIPT>
    var JavascriptVersion = "unbekannt";
    var JScript = false;           // Annahme: kein Microsoft

    function Anzeigen()
    {
        var Kette = "JScript: ";

        Kette += "Aktuelle Maschine = " + ScriptEngine();
        Kette += " mit Hauptversion " + ScriptEngineMajorVersion();
        Kette += " mit Unterversion " + ScriptEngineMinorVersion();
        Kette += " mit Buildnummer " + ScriptEngineBuildVersion();

        alert(Kette);
    }
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript">
    JavascriptVersion = "1.0";      // bitte keine var-Kodierung, da in älteren Versionen eine var-Anweisung unbekannt
ist
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript1.1">
    JavascriptVersion = "1.1";
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript1.2">
    JavascriptVersion = "1.2";
</SCRIPT>
<SCRIPT LANGUAGE="JScript">
    JScript = true;               // JScript kennt keine Sprachversion, dafür aber Versionen der Scriptmaschine,
                                // was Sinn macht, denn die Scriptmaschine bestimmt den Umfang
                                // von JScript und der Kompatibilität zum Javascript-Standard.
</SCRIPT>

<SCRIPT>
```



```

    if (JScript)                // if-Anweisung ist in allen Scriptversionen bekannt
    { Anzeigen(); }              // Funktionsaufruf ist in allen Scriptversionen möglich
    else
    { alert(JavascriptVersion); } // alert() ist in allen Scriptversionen bekannt
</SCRIPT>
</HEAD>

```

1.4. Feststellung der aktuellen JScript-Maschine

Es werden folgende Methoden verwendet, die **keine Objektreferenz** benötigen:

.ScriptEngine()	Sprache der gerade benutzten Scriptmaschine im Internet Explorer
	"JScript" Microsoft JScript
	"VBA" Microsoft Visual Basic for Applications
	"VBScript" Microsoft Visual Basic Scripting Edition
.ScriptEngineBuildVersion()	Buildnummer der gerade benutzten Scriptmaschine im Internet Explorer
.ScriptEngineMajorVersion()	Hauptversion der gerade benutzten Scriptmaschine im Internet Explorer
.ScriptEngineMinorVersion()	Unterversion der gerade benutzten Scriptmaschine im Internet Explorer

Beispiel:

```

function Anzeigen()
{
    var Kette = "";

    Kette += "Aktuelle Maschine = " + ScriptEngine();
    Kette += " mit Hauptversion " + ScriptEngineMajorVersion();
    Kette += " mit Unterversion " + ScriptEngineMinorVersion();
    Kette += " mit Buildnummer " + ScriptEngineBuildVersion();

    alert(Kette);
}

```

2. Javascript in HTML einbinden

2.1. Javascript-Direktkodierung in das HTML-Dokument

2.1.1. Javascript-Kodierung per HTML-Tag

Interpretation des HTML-Dokumentes von oben nach unten

Mit der Interpretation wird der Javascriptcode ausgeführt:

Javascript-Funktionen werden nur dann ausgeführt, wenn sie mit einer Argumentenliste "()" kodiert sind (egal ob volle oder leere Argumentenliste)

Kodierung einer Javascript-Funktion ohne "()" bewirkt Lieferung des **Zeigers** auf die Funktion und kein Funktionsaufruf

Javascriptcode innerhalb <HEAD> ... <HEAD> wird vor dem Javascriptcode innerhalb <BODY> ... </BODY> ausgeführt, da der HEAD-Abschnitt vor dem BODY-Abschnitt interpretiert wird.

Beispiel:

```

<HTML>
  <HEAD>
    .....
    <TITLE> ..... </TITLE>
    <SCRIPT>
      // hier die Scriptanweisungen
    </SCRIPT>
  </HEAD>
  <BODY>
    <SCRIPT>
      ....
    </SCRIPT>
    <NOSCRIPT> Text fuer Browser, die Script nicht kennen </NOSCRIPT>
    ....
  </BODY>
</HTML>

```

Empfehlung: LANGUAGE-Attribut in <SCRIPT ...> mitkodieren, wenn bestimmte Version verlangt wird

für Browser, der kein Script kennt, muss kodiert werden:

```

<SCRIPT ....>
<!--

```



```
// hier die Scriptanweisungen
-->
</SCRIPT>
```

wobei `<!---->` der mehrzeilige Kommentarbegrenzer ist

```
<NOSCRIPT>
text
</NOSCRIPT>
```

wobei Text nur dann angezeigt wird, wenn der Browser das `<NOSCRIPT>`-Tag kennt

2.1.2. Javascript-Kodierung als Wert eines HTML-Attributes im HTML-Tag (Javascript-Entity)

jedes beliebiges HTML-Attribut

Syntax:

```
attribut="&{javascript_ausdruck};"
```

javascript_ausdruck muss den Werte des Attributes liefern (Typengleichheit !!)

Beispiel für automatische Bildanpassung auf 25% der aktuellen Fensterbreite bei Netscape:

```
<IMG WIDTH="&{Math.ceil(self.innerWidth*0,25)};" ...>
```

2.1.3. Javascript-Kodierung als Aktion eines Eventhandlers im HTML-Tag (Javascript-Entity)

Syntax:

```
onXXX=" javascript:javascript_code"
onXXX=' javascript:javascript_code'
```

`" "` bzw `' '` muss kodiert werden

on Suffix für Eventhandler
muss kodiert werden

javascript: optional kodierbar

XXX für Bezeichner des Events
Gross-Klein egal

javascript_code beliebig

Beispiel 1:

```
<BODY onload="javascript:alert('Hallo!');">
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
    funktion Anzeige()
    {alert("Das Laden von BODY, also des Dokumentes, wurde soeben beendet !");}
</SCRIPT>
<BODY onload="javascript:Anzeige();">
</BODY>
</HTML>
```

2.1.4. Javascript-Kodierung als Aktion eines Eventhandlers im SCRIPT-Tag

ab IE 4.x

Das Script-Tag wird um die Attribute FOR und EVENT erweitert:

```
<SCRIPT FOR=objekt_bezeichner EVENT=event_bezeichner .....>
.....
</SCRIPT>
```

objekt_bezeichner Standard-Objekt laut DOM z.B. window
ID des instanziierten Objekt z.B. laut ID-Attribut
Kodierung wahlweise mit oder ohne `" "` bzw. `' '`

event_bezeichner alle Events **mit** Präfix on z.B. onload
siehe Objekt event
Kodierung wahlweise mit oder ohne `" "` bzw. `' '`



Ansonsten gilt das Übliche zum Script-Tag, also auch die Lage im HEAD und/oder BODY.

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT FOR=window
    EVENT=onload
>
    var Filter0 = ID_Div.filters[0];
    Filter0.Apply();
    ID_Div.innerHTML= "<IMG SRC='test.jpg' WIDTH=300 HEIGHT=300>";
    Filter0.Play();
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID= "ID_Div"
        STYLE= "position:absolute;width:300;height:300;top:20;left:20;
            filter:revealTrans(transition=12,duration=8)
        "
    >
        Dieser Text wird ueberblendet per Filter revealTrans (nur IE).
        <BR>
        Der Filter wird aktiv mit Laden des Dokumentes in das Fensters.
    </DIV>
</BODY>
</HTML>
```

2.1.5. Javascript-Kodierung zur Laufzeit des HTML-Dokumentes

Sämtlicher Scriptcode, der im Dokument abgearbeitet werden soll, muss **spätestens** zur Laufzeit und nicht unbedingt bereits beim **Laden** des HTML-Dokumentes vorliegen.

Mit Javascript ist der Programmierer in der Lage, Script mit speziellen Javascript-Anweisungen zu erzeugen, welche den erzeugten Scriptcode auch **sofort parsen und ausführen** und das auch **nach dem Laden** des HTML-Dokumentes, also zu seiner Laufzeit. Diese speziellen Anweisungen müssen allerdings bereits dann im Dokument vorliegen, wenn das Dokument geladen wird. Sie müssen also im Dokument kodiert und beim Laden bekannt sein. Der Aufruf dieser Methoden kann während oder nach dem Laden erfolgen.

Der Umfang des per Script zu erzeugenden Skriptcodes ist beliebig. Natürlich kann der zu erzeugende Scriptcode mit zu erzeugendem HTML-Code gemischt werden.

Im Falle der Erzeugung von HTML-Code ist es wichtig, mit welchen Methoden dieser erzeugt wird: Ob mit Methoden des HTML-DOM oder mit Methoden, die nicht HTML-DOM-konform sind (siehe weiter unten). Für die Anwendung letzterer Methoden im Internet Explorer ist unbedingt zu beachten, dass der HTML-Code bereits **während des Ladens** des HTML-Dokumentes, also **vor** der Auslösung des Ereignisses onload erzeugt werden muss (Begründung: siehe weiter unten).

Wenn der Programmierer den gesamten BODY-Teil des HTML-Dokumentes (inklusive der Tags <BODY> und </BODY>) per Script erzeugen will, so kann er dieses tun: Der Scriptcode muss dazu komplett im HEAD kodiert sein und wird mit dem Parsen des HEAD abgearbeitet, also vor dem Erreichen des Tags </HTML>. Natürlich wird ein hinter dem HEAD per BODY-Tag kodierter Body auch abgearbeitet, aber eben erst **nach** dem HEAD. Daher ist es empfehlenswert, entweder den Body nicht oder als leer zu kodieren (<BODY></BODY>).

Falls es der Browser zulässt, kann das **gesamte** HTML-Dokument **dynamisch erzeugt werden** und zwar inklusive der Werte zu Attributen von HTML-Elementen. Achtung: Ein User, der Javascript deaktiviert hat, wird dann nichts zusehen bekommen ! Im Übrigen ist das Feature, ein HTML-Dokument aus purem Script erstellen zu können, ein Analogon zu PHP, mit dem HTML-Code per PHP-Code erzeugt werden kann.

Skriptcode kann also

- dynamisch durch Script **verwaltet** werden
- sich auf Variablen, Funktionen und Objekte beziehen, die erst zur Laufzeit in Art und Umfang erzeugt werden sollen
- kann HTML-Code dynamisch verwalten.

Es gibt diverse und z.T. browserspezifische Anweisungen, die Scriptcode zur Laufzeit erzeugen. Nachfolgend werden 3 wichtige und oft benutzte Anweisungen beschrieben, die **weder** browser- und nicht objektspezifisch **noch** Methoden des DOM sind. Diese 3 Methoden erzeugen einen Datenstrom, der im Falle von zu erzeugendem HTML-Code wie das Laden einer HTML-Datei wirkt (siehe weiter unten). Hinweis: Das Laden eines Dokumentes aus einer HTML-Datei kann nur über einen Datenstrom durch Auslesen der Datei realisiert werden.

Für Programmierer unter Windows XP mit dem Internet Explorer bitte **unbedingt** beachten: Die Scriptmaschine unter Windows XP ist wesentlich **weniger fehlertolerant** als die Scriptmaschine unter Windows 98 bei identischer Browserversion und identischem Patch-Stand der Browsersoftware. Unter Windows XP ist der Internet Explorer 6.x implementiert. JScript-Code, der unter Windows 98 einwandfrei funktioniert, muss es unter Windows XP **nicht** ! Javascript-Code, der unter Windows XP funktioniert, wird es auch unter Windows 98 tun.



Mit anderen Worten: Die Scriptmaschine unter Windows XP ist bezüglich Fehler **nicht** abwärtskompatibel ! Deswegen bitte **unbedingt** den Scriptcode unter Windows XP testen !

2.1.5.1. Methode eval()

Diese Methode parst und führt den **Javascriptcode** in der Zeichenkette (String oder Stringliteral) sofort aus.
Der Javascriptcode darf **keinen** HTML-Code enthalten.
wird im Kontext des kodierten eval() ausgeführt.

Hinweis: Die Methoden document.write() und document.writeln() können Script- **und** HTML-Code ausführen.

Syntax:

```
eval(zeichenkette_mit_beliebigem_javascript_inhalt);
```

Die Zeichenkette kann zuvor mit den üblichen Kettenoperationen gebildet werden.
Es ist möglich, auch Ausdrücke zu verwenden, die aber immer String liefern müssen.

Beispiel:

```
eval("var Feld = new Array();"); // Die Variable Feld wird instanziiert !
```

2.1.5.2. Methoden document.write() und document.writeln()

Diese Methoden parsen und führen den **Javascriptcode** in der Zeichenkette (String oder Stringliteral) sofort aus.
Der Javascriptcode darf HTML-Code enthalten.
wird im Kontext des BODY-Teils vom **aktuellen** HTML-Dokument ausgeführt und zwar unabhängig davon,
wo die Anweisungen document.write() und document.writeln() kodiert sind
(für den Internet Explorer bitte **unbedingt** weiter unten lesen !).

Hinweis: Die Methode eval() kann nur Script- und **keinen** HTML-Code ausführen.

Syntax:

```
document.write(zeichenkette_mit_beliebigem_javascript_und_oder_html_inhalt);  
document.writeln(zeichenkette_mit_beliebigem_javascript_und_oder_html_inhalt);
```

Die Zeichenkette kann zuvor mit den üblichen Kettenoperationen gebildet werden.
Es ist möglich, auch Ausdrücke zu verwenden, die aber immer String liefern müssen.

document.writeln() erzeugt automatisch ein
 (Zeilenumbruch)
Ein
 kann auch durch "\n" kodiert werden z.B. bei document.write()

Beispiel 1 für dynamisches Zeigerfeld auf OBJECT-Elemente:

```
var ObjektZeigerFeld = new Array();           // dynamisches Feld

function ObjektErzeugen(ObjektNummer)
// Die Objektnummer muss >=0 sein
// eine lückenlose Vergabe ist nicht nötig
{
    // externes Objekt einbinden, wobei der Name unbedingt mit der Objektnummer versehen wird
    document.write(      '<OBJECT ID="OBJECT_ID" + ObjektNummer.toString() + "></OBJECT>');

    // Zeiger vom externen Objekt im ObjektZeigerFeld[] merken
    document.write(      '<SCRIPT LANGUAGE="JavaScript1.2">'
        + 'ObjektZeigerFeld[' + ObjektNummer.toString()
        + ']=document.OBJECT_ID + ObjektNummer.toString() + '\n'
        + '</SCRIPT>'
    );
}

for ( var i=0; i<5; i++)
{
    // Lücke lassen bei Index 2 bzw. 3 bzw. 4
    if ( (i==0)
        || (i==1)
        || (i==5)
    )
    {ObjektErzeugen(i);}           // ID-Attribut bekommt den Wert      "OBJECT_ID0"
                                //                                     "OBJECT_ID01"
                                //                                     "OBJECT_ID5"
}

Die Zeiger sind per  ObjektZeigerFeld[0] bzw. OBJECT_ID0
                   ObjektZeigerFeld[1] bzw. OBJECT_ID1
                   ObjektZeigerFeld[5] bzw. OBJECT_ID5
```



ansprechbar.

Die Attribute der OBJECT-Tags sind noch anzupassen.

Beispiel 2 für dynamisches Zeigerfeld auf OBJECT-Elemente mit eigenen Script-Funktionen per Prototyping:

```
// Funktion als String
var RumpfCodeDerFunktion1 =      'if (Wert1 != 0)\n'
                                +  '{alert("Wert1 ist " + Wert1);}\n'
                                +  'else\n'
                                +  '{alert("Wert1 ist Null !");}\n';

// internes Zeigerfeld für Objekte
var ObjektZeigerFeld = new Array();      // dynamisches Feld
                                           // Es wird die Eigenschaft .prototype erzeugt

function ObjektErzeugen(ObjektNummer)
// Die Objektnummer muss >=0 sein
// eine lückenlose Vergabe ist nicht nötig
{
    //      externes Objekt einbinden, wobei der Name unbedingt mit der Objektnummer versehen wird
    document.write(      '<OBJECT ID="OBJECT_ID" + ObjektNummer.toString() + "></OBJECT>');

    //      Zeiger vom externen Objekt im ObjektZeigerFeld[] merken
    document.write(      '<SCRIPT LANGUAGE="JavaScript1.2">'
                        +  'ObjektZeigerFeld[' + ObjektNummer.toString()
                        +  ']=document.OBJECT_ID' + ObjektNummer.toString() + ';\n'
                        +  '</SCRIPT>'
                        );

    //      globale Kodierung der objekt eigenen Methode mit einem objektinternen Bezeichner, der
    //      eindeutig die Objektzugehörigkeit anzeigt
    //      die Funktion wird zugleich instanziiert und ist somit per Zeiger adressierbar
    document.write(      '<SCRIPT LANGUAGE="JavaScript1.2">'
                        +  'function OBJECT_ID' + ObjektNummer.toString() + '_Funktion1()\n'
                        +  'RumpfCodeDerFunktion1'
                        +  '</SCRIPT>'
                        );

    // Erweiterung des Objektes per Prototyping um      die Funktion1 durch deren Zeiger
    //                                                    den Wert1, der zugleich instanziiert und initialisiert wird
    document.write(      '<SCRIPT LANGUAGE="JavaScript1.2">'
                        +  'ObjektZeigerFeld[' + ObjektNummer.toString()
                        +  '].prototype.Funktion1= OBJECT_ID' + ObjektNummer.toString() + '_Funktion1\n'

                        +  'ObjektZeigerFeld[' + ObjektNummer.toString()
                        +  '].prototype.Wert1= 0; // Initialisierung\n'
                        +  '</SCRIPT>'
                        );
}
```

Beim Prototyping von Funktion1 bitte nicht () kodieren, da ja ein Zeiger übergeben werden soll !

Der Zugriff auf die Daten und Methoden des Objektes erfolgt über ObjektZeigerFeld[ObjektNummer].

z.B. ObjektZeigerFeld[ObjektNummer].Funktion1();
 ObjektZeigerFeld[ObjektNummer].Wert1=33;

Die objekt eigene Funktion1 kann natürlich auch Parameter haben.

Die Doppelbezeichnung ein und desselben Funktionscodes mit

 "Funktion1"
 und "OBJECT_IDx_Funktion1" x ist die jeweilige Objektnummer

macht Sinn, sobald mehrere Instanzen des externen Objektes gebildet werden sollen, da somit alle Objekte den selben Funktionsbezeichner haben, aber intern namentlich verschiedene. **Und:** Die Objekte haben je ihre **eigene** Funktion, die von keinem anderen Objekt benutzt werden kann. Das Prinzip der Kapselung wurde somit erfüllt. Der Funktionsname ist für den Programmierer einheitlich. Durch den Bezug per




```
ObjektZeigerFeld[ObjektNummer].Funktion1
ObjektZeigerFeld[ObjektNummer]Wert1
```

wird die Kapselung aufrechterhalten.

Diese Methode kostet Ressourcen und bläht die HTML-Datei auf ! Daher ist es natürlich auch möglich, nur 1 Methode zu deklarieren und dann den Objekten den identischen Zeiger zuzuweisen. Wichtig ist dabei, dass die Objekte dann **nicht parallel** auf diese Funktion zugreifen dürfen, da die Funktion in keinem Objekt gekapselt ist !!!

Durch die Auslagerung des Code der Funktion1 in die Stringvariable

```
RumpfCodeDerFunktion1
```

wird Übersicht erzeugt. Man beachte, dass
" und ' sich nicht inkorrekt mischen !
die maximal mögliche Gesamtlänge des Strings beachtet werden muss, also eventuell mehrere Variablen zu kodieren sind.

Durch die Auslagerung der Variable

```
RumpfCode DerFunktion1
```

in eine eigene JS-Datei und deren Einbindung in das HTML-Dokument, kann der Aufbau der Funktion1, also des Objektes, von außen gesteuert werden, ohne das HTML-Dokument editieren zu müssen.

Empfehlenswert ist es, per \n am Zeilenende einen Zeilenumbruch zu erzeugen, damit die jeweilige erzeugte Zeile mit Ausführung von document.write() nicht unnötig zu lang wird, da Interpreter von Browsern nicht unbedingt überlange Zeilen ausführen können.

Beim Internet Explorer ist unbedingt zu beachten:

Die Ausführung von document.write() bzw. document.writeln(), das **HTML-Tags** erzeugt, hat 2 Konsequenzen und zwar genau dann, wenn diese Anweisung **nach dem kompletten Laden des Dokumentes, also nach Auslösung des Ereignisses onload** aktiviert wird:

Fakt 1:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen HTML-Dokumentes**, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden **keine** des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

Die Methoden write() und writeln() erzeugen einen Datenstrom aus HTML-Elementen und das neue Dokument empfängt diesen so, als würde er aus einer HTML-Datei stammen. Mit Ende des Datenstromes wird quasi ein Dateiende erkannt und das neue Dokument löst das Ereignis onload aus. Damit wird aber das neue Dokument zum aktuellen Dokument, also im Fenster über dem des alten Dokumentes angezeigt. Da das Fenster des neuen Dokumentes automatisch erzeugt wurde (nicht per Methode open()), sind also die Standards für eine Fenstererzeugung verwendet worden. Damit hat das neue Dokument einen History-Eintrag. Der User kann nun mit diesem zwischen dem alten und neuen Dokument umschalten.

Fakt 2:

Im Falle der o.g. nachträglichen Veränderung des Dokumentes um HTML-Elemente per write() bzw. writeln() kennt das neue, automatisch geöffnete Dokument das alte Dokument nur **als Eltern**. Es muss also im neuen Dokument mit dem Zeiger auf die Eltern gearbeitet werden, wenn Daten und Routinen der Eltern benutzt werden sollen (siehe Objekt window bzw. Objekt document). Mit anderen Worten: Das neue Dokument muss dann komplett per Script erzeugt werden, denn dieser Zeiger lässt sich nur über Script ansprechen. Jedes geladene Dokument hat ansonsten seine eigenständige Umgebung.

2.2. Javascript-Kodierung in externer Datei *.js (Javascript-Code mehrfach verwenden)

ab Javascript 1.1

ab JScript, das Javascript 1.1 unterstützt

Dateiname: möglichst 8 Zeichen
 alles klein
 frei wählbar
 Suffix *.js

Datei: ASCII
 darf **nur** Javascript-Code und keinen HTML-Code enthalten, also auch kein <SCRIPT> </SCRIPT>



Mimetyt "text/javascript" muss mit dem Suffix von *.js verknüpft sein

Einbinden im HEAD des HTML-Dokument per eigenem <SCRIPT></SCRIPT>

```
<SCRIPT LANGUAGE="....."
      TYPE="text/javascript"
      SRC="script_datei.js"
>
// alle andersartigen Javascript-Anweisungen werden hier vom Browser ignoriert !
</SCRIPT>
```

SRC: Dateiname auch mit Pfad möglich
mehrere SRC sind möglich --> Empfehlung: pro JS-Datei das eigene <SCRIPT> </SCRIPT>

<SCRIPT> ... </SCRIPT> innerhalb </HEAD> .. </HEAD> und <BODY> ... </BODY> kodierbar

Bei Frames: Trotz gemeinsamer Quelle (js-Datei) wird der in allen Frames identische Bezug auf ein Javascript-Element immer im Kontext des Frame ausgeführt (pro Frame eigenständig).

Beispiel:

Inhalt von TEST.HTM:

```
<SCRIPT LANGUAGE="....."
      TYPE="text/javascript"
      SRC="test.js"
>
      // alle andersartigen Javascript-Anweisungen werden hier vom Browser ignoriert !
      alert(TestWert); // wird nicht ausgeführt
</SCRIPT>

<SCRIPT>
      var Kette = "TestWert";
      alert(Kette + " " + TestWert);     // TestWert 10 wird angezeigt
</SCRIPT>
```

Inhalt von TEST.JS:

```
var TestWert=10;
```

Beispiel für Einbindung von Datensätze als Teil einer Javascript Datei (*.js):

daten.js

Pro Satz sind 3 Felder vorhanden, die mit Zeichenkettenwerten belegt werden.
Die Sätze werden in eine Array-Variablen abgelegt → Array-Indizierung ab 0 !!

Aufbau von daten.js:

```
var anzahl_saetze=3;                             /* Anzahl stets ab 0, sonst beliebig */

function erzeuge_satz(satz_teil0, satz_teil1, satz_teil2)
                                                   /* Konstruktor zur satzweisen Feldbelegung */
{
    this.satz_teil0=wert0;
    this.satz_teil1=wert1;
    this.satz_teil2=wert3;
}

datensaetze_feld=new Array(anzahl_saetze);
datensaetze_feld[0]=new erzeuge_satz("0_1","0_2","0_3");
datensaetze_feld[1]=new erzeuge_satz("1_1","1_2","1_3");
datensaetze_feld[2]=new erzeuge_satz("2_1","2_2","2_3");
```

Einbindung von daten.js

Mit Einbinden der *.js-Datei wird der Javascript-Code SOFORT interpretiert. Damit werden alle Anweisungen außerhalb von Funktionen sofort ausgeführt.

```
<HTML>
<HEAD>
```



```

<SCRIPT LANGUAGE="JavaScript" SRC="daten.js"> </SCRIPT>

<SCRIPT LANGUAGE="JavaScript" >
<!--
    // Es wird im satz_teil0 eines jeden Satzes gesucht.
    // Das Stichwort ist der Suchbegriff.
    // Im Satz und im Suchbegriff können Gross -und kleinbuchstaben auftauchen.
    // Der Suchbegriff kann im satz_teil0 als Teilkette auftauchen, aber auch das gesamte Feld umfassen.

    function vergleich(satznr,suchwort)
    {
        suchwort_laenge=suchwort.length;           /* Suchwort akt. Länge */
        datensatz_laenge=datensaetze[satznr].satz_teil0.length;
                                                    /* Feld 0 im Satz: akt. Länge */
        if (datensatz_laenge >= suchwort_laenge)
        {
            anzahl_pos= datensatz_laenge - suchwort_laenge;

            for ( var pos=0; pos <= anzahl_pos; pos++)
            {
                teilkette =
                datensaetze[satznr].satz_teil0.substring(pos,suchwort_laenge).toLowerCase();

                if (teilkette == suchwort.toLowerCase())
                {return true;}           // gefunden im satz_teil0 des Datensatzes
            }
        }
        else {return false;}           // nicht gefunden im satz_teil0 des Datensatzes
    }

    function suche(suchwort)
    {
        var gefunden=false;

        if (anzahl_saetze > 0)           // anzahl_saetze laut js-Datei !!
        {
            var satzzahler=-1;
            while ( (!gefunden)
                && (satznr <= anzahl_saetze)
            )
            {
                satzzahler++;           // um 1 erhöhen
                gefunden= vergleich(satzzahler,suchwort);
            }
        }

        return gefunden;
    }

    function starten()
    {
        suchwort=document.form_name.input_name.value;

        if (suchwort == "")
        {alert("Bitte Suchwort eingeben !");}
        else
        {
            if (suche(suchwort))
            {alert(" gefunden !");}
            else
            {alert("nicht gefunden!");}
        }
    }
-->
</SCRIPT>
</HEAD>
<BODY>
    <FORM NAME="form_name" onSubmit="starten()">
        Bitte Suchbegriff eingeben:
        <INPUT NAME="input_name" TYPE=TEXT>           </INPUT>
        <INPUT TYPE=submit VALUE="Suche starten"> </INPUT>

```



```

</FORM>
</BODY>
</HTML>

```

2.3. Javascript und Browserperformance

Die Browser-Performance zur Darstellung des HTML-Dokumentes oder eines Objektes kann wie folgt erhöht werden:

In HTML: Wenn das ID-Attribut zum Tag zulässig ist, dann immer kodieren.
 Wenn das NAME-Attribut zum Tag zulässig ist, dann immer kodieren.
 Wenn NAME- und ID-Attribut zum Tag zulässig sind, dann ID-Attribut kodieren,
 es sei denn, es wird eine Referenz über das NAME-Attribut benötigt.

In Script:

Bei Referenz auf eine Methode oder Eigenschaften eines Objektes sollte immer **zuerst** ein Zeiger auf das Objekt erzeugt werden, um **dann** mit diesem Zeiger die Methode bzw. Eigenschaft anzusprechen. Das gilt **vor allem** für Objekte, die Kinder haben, welche aufgerufen werden sollen. Je **länger** die Punktnotation ist, um so mehr Aufwand hat der Browser beim Interpretieren, denn für jede Ebene in der Notation muss der Zeiger der Ebene berechnet werden.

Die Zeigerbildung schafft auch Übersichtlichkeit im Quellcode.

```

Bsp.:    var BodyObjekt = document.body;
         BodyObjekt.eigenschaft = ....;

         // Die langsamere Variante ist
         //    document.body.eigenschaft = ...;

Bsp. für IE:
var FeldDerZeigerAllerHTMLElementeImDokument = document.all;

var ZeigerAufHTMLObjekt =
    FeldDerZeigerAllerHTMLElementeImDokument.IDdesHTMLElementes;

ZeigerAufHTMLObjekt.eigenschaft = ....;

// ID des HTML-Elementes    z.B. laut ID-Attribut im HTML-Tag

```

Die Zeigerbildung ist auch dann sinnvoll, wenn das betroffene Objekt mehrmals im Scriptcode referenziert werden soll.

```

Bsp. für IE:
var FeldDerZeigerAllerHTMLElementeImDokument = document.all;

var FeldDerH1TagsImDokument = FeldDerZeigerAllerHTMLElementeImDokument.tags("H1");

var AnzahlDerH1TagsImDokument = FeldDerH1TagsImDokument.length;

for (var Zahler = 0; Zahler < AnzahlDerH1TagsImDokument; Zahler++)
{ ..... }

// Die langsamere Variante ist
//    for (var Zahler = 0; Zahler < document.all.tags("H1").length; Zahler++)
//    { ..... }

//    pro Vergleich mit Zahler muss document.all.tags("H1").length immer neu
//    berechnet werden

```

Ein Objekt sollte immer mit **absolutem Pfad** referenziert werden, der bereits vorher per Teilzeiger referenziert sein sollte, falls der Teilzeiger in Mehrfachkodierungen verwendet werden kann.

```

Bsp.:    var Kette = window.navigator.userAgent; // Zeigers des Fensters explizit kodieren,
                                                //    also window als Pfadursprung

         // Die langsamere Variante ist
         //    var Kette = navigator.userAgent;
         //    Zeiger auf window muss impliziert werden

```

2.4. Javascript- und HTML-Dateien auf dem Server

Javascript-Dateien sollten mit ihren zugehörigen Dokumenten auf dem Server in systematischer Struktur abgelegt sein, die dem Aufbau des HTML-Projektes (Web-Projektes) als Gesamtheit aller HTML-Komponenten entspricht. Die Struktur auf dem Server entspricht auch einer Baumstruktur wie im Dateisystem von Windows (nur dass möglichst Dateinamen in der 8.3-Kodierung analog zu DOS verwendet werden sollten z.B. t2345678.htm und nicht lt345678.htm und nicht T2345678.htm und nicht TestDateiMitHTMLCode.htm).



2.4.1. robots.txt und Suchmaschinen

Die Scanvorgänge von Suchmaschinen lassen sich beeinflussen über

META-Tag
ASCII-Datei robots.txt

Der Nutzung von Meta-Tags ist die Anwendung von robots.txt vorzuziehen.

2.4.1.1. robots.txt und ihre Lage auf dem Server

NUR im Hauptverzeichnis des Webs auf dem Server

Bsp. www.test.de/robots.txt

Dateinamen klein schreiben
als Pfadtrenner nur / und nicht \ kodieren
sobald sich diese Datei auf dem Server befindet, ist sie wirksam

2.4.1.2. robots.txt die nicht auf dem Server vorhanden ist (Scan-Standard)

Existiert keine robots.txt im Hauptverzeichnis, so gilt folgender Scan-Standard:

JEDE Suchmaschine darf alles (auch Dateien etc.) absキャン.

2.4.1.3. robots.txt und Aufbau**2.4.1.3.1. robots.txt als Zeilenfolge-Script erstellen**

als reine ASCII-Text, also z.B. mit Notepad
keine Formate etc. wie z.B. unter WORD !!

jede Zeile per Return-Taste-Eingabe beenden

Kodierung ist in ihrer nachfolgend gezeigten Groß-Kleinschreibung einzuhalten

2.4.1.3.2. robots.txt und ihre Blöcke

Block: Folge von Zeilen, die genau eine Scan-Variante beschreiben

beliebige Anzahl von Blöcken sind in robots.txt möglich

Trennung von Blöcken erfolgt nicht, da jeder neue Block an seiner 1. Zeile erkannt wird

2.4.1.3.2.1. robots.txt-Block: Aufbau Zeile 1 (Zulassen bzw. Sperren von Suchmaschinen)

Diese Zeile dient zum Ein- bzw. Ausschließen von Suchmaschinen für den Scan des HTML-Dokumentes

2.4.1.3.2.1.1. robots.txt und Zeile 1: ALLE Suchmaschinen dürfen scannen

user-agent: *

Hinweis: zwischen Doppelpunkt und * bzw. Name der Suchmaschine ist ein Leerzeichen zu kodieren

2.4.1.3.2.1.2. robots.txt und Zeile 1: KEINE Suchmaschine darf scannen

Diese Variante ist nicht kodierbar.

2.4.1.3.2.1.3. robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf scannen

user-agent: WebCrawler

WebCrawler darf scannen

Sollen mehrere Suchmaschinen ausgewählt werden, so muss pro Suchmaschine ein eigener Block mit Zeile 1 wie im Beispiel kodiert werden.
Der Name der Suchmaschine muss dafür exakt bekannt sein.

Hinweis: zwischen Doppelpunkt und * bzw. Name der Suchmaschine ist ein Leerzeichen zu kodieren

2.4.1.3.2.1.4. robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf NICHT scannen

user-agent: WebCrawler

WebCrawler darf scannen, aber **ab Zeile 2** sind alle Inhalte als gesperrt zu markieren

Sollen mehrere Suchmaschinen ausgewählt werden, so muss pro Suchmaschine ein eigener Block mit Zeile 1 wie im Beispiel kodiert werden.
Der Name der Suchmaschine muss dafür exakt bekannt sein.

Hinweis: zwischen Doppelpunkt und * bzw. Name der Suchmaschine ist ein Leerzeichen zu kodieren

2.4.1.3.2.2. robots.txt-Block: Aufbau ab Zeile 2 (Zulassen bzw. Sperren von Inhalten auf dem Server für Suchmaschinen, die scannen)

2.4.1.3.2.2.1. robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen gescannt werden**Disallow:**es muss **nur noch diese Zeile** kodiert werden

Daten: Haupt- und alle Unterverzeichnisse und alle HTML-Dateien mit deren Bildern etc.

Für folgende Fälle ist diese Variante nicht zulässig:

Verzeichnisse, die mit User-Passwort geschützt sein sollen
 Verzeichnisse, die private Dinge des Web-Programmierers enthalten
 ausgewählte Dateien wie HTML-Dateien die niemanden was angehen

2.4.1.3.2.2.2. robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen NICHT gescannt werden**Disallow: /**

sperrt das Hauptverzeichnis und damit alle Unterverzeichnisse

Hinweis: zwischen Doppelpunkt und / ist ein Leerzeichen zu kodieren

2.4.1.3.2.2.3. robots.txt und ab Zeile 2: Bestimmte Daten auf dem Server dürfen NICHT gescannt werden**2.4.1.3.2.2.3.1. robots.txt und ab Zeile 2: Bestimmte Verzeichnisse auf dem Server dürfen NICHT gescannt werden****Disallow: /pfad_angabe/**

pfad_angabe: Pfadkodierung mit / für dasjenige Verzeichnis, das nicht gescannt werden darf
 abschließendes / nicht vergessen !

pro Verzeichnis ist jeweils eine eigene Zeile zu kodieren

Hinweis: zwischen Doppelpunkt und dem ersten / ist ein Leerzeichen zu kodieren

2.4.1.3.2.2.3.2. robots.txt und ab Zeile 2: Bestimmte HTML-Dateien auf dem Server dürfen NICHT gescannt werden**Disallow: /pfad_angabe/datei_angabe**

pfad_angabe/dateiangabe: Pfadkodierung mit / für diejenige Datei, die nicht gescannt werden darf

pro Datei ist jeweils eine eigene Zeile zu kodieren

Hinweis: zwischen Doppelpunkt und dem ersten / ist ein Leerzeichen zu kodieren

Diese Kodierungsvariante ist wichtig für PHP-Skripte und Dateien, die Passwörter verwalten bzw. enthalten.

Sollte die Passwortverwaltung durch den Provider selbst angeboten werden, so prüfen, ob das vorgegebene Verzeichnis mit seinen
 Dateien vor dem Scannen geschützt sind !

2.4.1.3.2.2.4. robots.txt und ab Zeile 2: NUR bestimmte Daten auf dem Server dürfen gescannt werden

Diese Variante ist nicht kodierbar.

2.4.1.3.3. robots.txt und Beispiele

Beispiel: WebCrawler darf ALLES scannen

user-agent: WebCrawler**Disallow:**

Beispiel: WebCrawler darf NICHTS scannen

user-agent: WebCrawler**Disallow: /**

Beispiel: WebCrawler darf ALLES außer /test/ scannen

user-agent: WebCrawler**Disallow: /test/**

Beispiel: WebCrawler darf ALLES außer /test/test.htm scannen

user-agent: WebCrawler**Disallow: /test/test.htm**

Beispiel: ALLE Suchmaschinen dürfen ALLES scannen (robots.txt kann entfallen)

user-agent: ***Disallow:**

Beispiel: ALLE Suchmaschinen dürfen NICHTS scannen

user-agent: ***Disallow: /**

Beispiel: ALLE Suchmaschinen dürfen ALLES außer /temp/ scannen

user-agent: *

Disallow: /temp/

Beispiel: ALLE Suchmaschinen dürfen ALLES außer /test/test.htm scannen

user-agent: *

Disallow: /test/test.htm

Beispiel: ALLE Suchmaschinen dürfen ALLES außer /test/test.htm UND /temp/ scannen

user-agent: *

Disallow: /test/test.htm

Disallow: /temp/

2.4.2. .htaccess und .htpasswd und Passwortschutz

Der Schutz von Dateien in einem Ordner auf dem Server ist Sache des Inhabers des Web-Projektes **und nicht Sache** des Serverinhabers. Es steht dem Inhaber des Web-Projektes frei, **seine** Daten zu schützen.

FTP-Zugriffsschutz:

Browser sind in der Lage, auch FTP-Ordner anzuzeigen und auf diese zuzugreifen. Manche Web-Anbieter nutzen Internet-Adressen mit ftp://.... **anstelle von** http://.... als öffentliche Systeme, die standardgemäß per FTP-Protokoll les- und schreibbar sind, wenn der Inhaber der FTP-Ordner keinen FTP-Schutz installiert hat. Existiert ein Zugriffsschutz, so erscheint mit dem Versuch eines Zugriffs im Browser ein Login-Fenster, das diverse Kennungen verlangt. Diese Kennungen müssen zwischen dem User, der per FTP zugreifen will, und dem Inhaber der FTP-Ordner **vereinbart** sein.

Das Web-Projekt ist standardgemäß per FTP auch schreibbar (FTP-Upload auf den Server), wenn nicht ein FTP-Zugriffsschutz vereinbart wurde: Für den Schutz des FTP-Upload auf den Server besitzt der Inhaber des Web-Projektes ein Passwort, dass dem Serverinhaber-Inhaber bekannt ist. Dieses Passwort liegt in einer Datenbank des Serverinhabers und wird mit der Passwort-Eingabe des Inhaber des Web-Projektes abgeglichen, sobald ein FTP-Zugriff versucht wird. Die Datenbank des Servers ist durch den Inhaber des Web-Projektes nicht einsehbar, da der Serverinhaber Komponenten benutzt, die zugleich ein Login-Interface demjenigen bereitstellt, der FTP aktivieren will. Diese Komponenten werden server-lokal abgearbeitet und gelangen nie auf den PC des FTP-Users (außer das Login-Interface).

Der FTP-Schutz hat nichts mit dem Passwortschutz per .htaccess und .htpasswd zu tun.

Passwort-Schutz per .htaccess und .htpasswd:

Es können Ordner des Web-Projektes derart geschützt werden, dass ein User nur einen **erlaubten Zugriff** auf Webseiten bekommt. Standardgemäß sind Webseiten immer lesbar, um auf den Client (User-PC) übertragbar zu sein, damit sie der User auch im Browser anschauen kann. Mit Aktivierung des Passwortschutzes ist ein User-individueller Zugriff installiert, der auch für den Inhaber des Web-Projektes gilt. Der Inhaber kann pro User ein Passwort vergeben. Ein User **ohne Passwort** bekommt keinen Lesezugriff und kann damit den Inhalt des Ordners im Browser nicht anzeigen lassen.

Schreibzugriff auf Ordner kann nur per FTP erfolgen (siehe oben).

Um Dateien im Ordner auf dem Server vor unberechtigtem Zugriff zu schützen, müssen **pro zu schützenden Ordner** die Dateien .htaccess **und** .htpasswd mit ordnerspezifischen Inhalten hinterlegt werden., wobei der jeweilige Dateiname in Kleinbuchstaben **und mit führendem Punkt** kodiert sein muss. Des weiteren muss der Serverinhaber die Verarbeitung dieser Dateien durch den Server unterstützen. Sollte das der Fall sein **und** existieren beide Dateien im jeweiligen zu schützenden Ordner, so ist der Passwortschutz für diese Ordner sofort aktiv.

Die Dateien **.htaccess** und **.htpasswd** sind reine ASCII-Dateien und sollten z.B. nur mit NOTEPAD.EXE erstellt werden, wobei Formatierungen wie bei Word oder Wordpad und deutsche Umlaute und Sonderzeichen nicht verwendet werden dürfen.

Aufbau der Datei .htaccess

```
AuthType Basic
AuthName "freier Text"
AuthUserFile absoluter_pfad/Ordner/.htpasswd
<Limit GET POST PUT>
require user freier_username_1
....
require user freier_username_n
</Limit>
```

freier Text als Hinweis
 muss in " " kodiert sein

absoluter_pfad auf dem Server
 immer von der Root aus
 Bsp.:



/root/xx/yy/Ordner/.htpasswd

mit root als Platzhalter für die Wurzel der Baumstruktur, welche frei benannt sein kann
xx und yy als Platzhalter für frei benennbare Ordner im Pfad
Ordner als Platzhalter für den frei benennbaren und zu schützenden Ordner

Achtung: Gross-Klein wird unterschieden, also die Namen der Pfadelemente korrekt kodieren !

freier_username_1 bis freier_username_n pro Username genau 1 Zeile mit **require user**

z.B. otto_1 bis otto_n

.... als symbolischer und nicht zu kodierender Platzhalter für die User ab 2 bis n

Aufbau der Datei .htpasswd

freier_username_1:verschlussetes_password

....

freier_username_n:verschlussetes_password

pro User laut Zeilen

require user freier_username_1

....

require user freier_username_n

in der Datei .htaccess

genau 1 Zeile in der Datei .htpasswd

z.B. otto1:Ho/87YN6YqP89

Doppelpunkt muss kodiert werden

.... als symbolischer und nicht zu kodierender Platzhalter für die User ab 2 bis n

Die Passwortverschlüsselung muss anhand einer standardisierten Formel erfolgen, die diverse Anbieter als online-bediensbares Programms anbieten (eventuell der Serverinhaber selbst):

z.B. <http://www.linux-profis.de/php4>
<http://www.inch.com/info/tech/HOWTOS/htaccess/htpasswd.html>

Es ist Sache des Inhabers des Web-Projektes, sich die Verschlüsselung zu besorgen.

3. Javascript-Elemente (Auswahl)

Kodierungsregeln:

Unterscheidung Groß von Klein für Anweisungen
Variablen
Objekte etc.

Es gibt KEINE vereinheitlichte Kodierungsregeln wie bei einer compilerbedingten Programmiersprache. Die Scriptmaschine als Javascript-Interpreter ist mehr oder weniger Fehlertolerant (je nach Hersteller).

Empfehlung: **erst** mit Kleinschreibung probieren, **dann** mit gemischter Kodierung versuchen.
Genau Syntaxkenntnisse

Hinweis für Kodierung eines HTML-Tags:

keine Unterscheidung zwischen Groß und Klein
Empfehlung: TAG-Bezeichner groß
TAG-Attribute groß
vordefinierte Werte für Attribute klein

3.1. Javascript-Bezeichner

Kodierungsregeln:

max. 32 Zeichen
nur Buchstabe, Ziffer, Unterstrich
Groß und Klein werden unterschieden
wenn durch Programmierer erzeugt, so kann ein reservierter Bezeichner **mit** enthalten sein



3.2. Javascript-Kommentar

im HTML-Code:

```
<!-- .... -->
```

bewirkt **nur**, dass Browser, der Script nicht kennt (also auch das SCRIPT-Tag nicht kennt), den Scriptcode als Kommentar überliest
mehrzeilig möglich

Beispiel: <SCRIPT>
<!--
function Test()
{alert("Test");}
-->
</SCRIPT>

im Scriptcode:

```
// das ist ein einzeliger Kommentar
```

```
/* das ist  
ein ein- oder mehrzeiliger  
Kommentar  
*/
```

Empfehlung: /* und */ immer untereinander kodieren
möglichst // verwenden anstelle /* */ also zeilenweise kommentieren, da jede Scriptanweisung möglichst ihre eigene Zeile bekommen sollte

Beispiele für Kombination von /* */ mit //

Beispiel 1: <SCRIPT>
<!--
function Test() // Eine Funktion
{alert("Test");} /* Die Funktion
zeigt in der Alert-Box
den Text an */
-->
</SCRIPT>

Beispiel 2: <SCRIPT>
<!--
function Test() // Eine Funktion
{alert("Test")} // ;} **Dieser Kommentar erzeugt Syntaxfehler**
/* Die Funktion
zeigt in der Alert-Box
den Text an */
-->
</SCRIPT>

Beispiel 3: <SCRIPT>
<!--
function Test() // Eine Funktion
{alert("Test") // ;} Dieser Kommentar erzeugt **keinen** Syntaxfehler
;} /* Die Funktion
zeigt in der Alert-Box
den Text an */
// /* wird nicht erkannt
/ // erzeugt Syntaxfehler, da / fehlt
-->
</SCRIPT>

Beispiel 4: <SCRIPT>
<!--
/*
function Test() // Eine Funktion
{alert("Test");} /* Die Funktion
zeigt in der Alert-Box
den Text an */
*/ // **die Funktion wird nicht erkannt**
-->
</SCRIPT>

siehe auch bedingtes Parsen nur beim Internet Explorer z.B. per @if
siehe auch Objekt comment



3.3. Datentypen

Der Datentyp legt fest, wie der Browser und der Programmierer
eine einer Instanz
den Wert der Instanz (z.B. Variable)
zu behandeln haben.

Es gibt Basis-Datentypen und Basis-Datenstrukturen, die in der Scriptmaschine implementiert sind.
Der Programmierer kann anhand dieser Basisgrößen neue Datentypen kreieren.

Hinweis: Script-Objekte, die Datentypen definieren, werden in den Beschreibungen zu den Objekten erklärt.

Die Deklaration einer Instanz ohne Datentyp kann erfolgen per `var Instanz;`
wobei mit der nächsten Wertzuweisung ein Daten-Typ automatisch zugewiesen und damit eine Adresse der Instanz erzeugt werden.

Eine Instanz ohne Datentyp
kann nur für eine Wertzuweisung auf die Instanz verwendet werden,
kann vom Browser nur als Platzhalter reserviert werden,
hat den Zeigerwert null (nicht numerisch 0).

Der Datentyp kann auch durch Kontextanalyse des Scriptcodes ermittelt z.B. beim Literal.

Ermittlung eines Basis-Datentyps bzw. einer Basis-Datenstruktur per `typeof`:

Syntax: `typeof operand;`

liefert String z.B.

```
"undefined"
"function"
"object"
"number"
"boolean"
"string"
```

Beispiel: `typeof 42` ergibt "number"

Die Datentyp-Konvertierung erfolgt .B. während der Wertzuweisung :
z.T. automatisch anhand von objektübergreifenden Methoden, die allen Objekten innewohnen,
per Methoden zum Basis-Datentyp bzw. Basis-Datenstruktur,
bei vom Programmierer kreierten Datentypen nur durch passend-programmierte Routinen.

Die Datentyp-Konvertierung kann folgende Werte liefern:

NaN	Not a Number Wert wird geliefert bei numerischer Operation mit nicht-numerischen Wert z.B. Zeichenkette bei Konvertierung von String nach numerisch, wenn String nicht Zeichen laut number Datentyp enthält
Infinity	Wert wird geliefert bei numerischer Operation, dessen Wertebereich den von Script überschreitet
-Infinity	Wert wird geliefert bei numerischer Operation, dessen Wertebereich den von Script überschreitet

Datentypen die gleichzeitig numerisch sind:

Typ	entspricht numerisch numerischem Wert
-----	-----
null-Zeiger bzw. undefined	jeder (egal welcher)
true	> 0
false	0
Leerkette UND Kette nur aus Leerzeichen	0
// aber jede andere Kette	> 0

Hinweis: Browser meldet undefined, wenn null-Zeiger erkannt.

3.3.1. array Datentyp (Basis-Datenstruktur)

basiert auf dem Script-Objekt Array
z.B. verwendet für Feld aus Literalen

Syntax für Feldfüllung eines instanziierten Feldes:

```
Zeiger = [ [ liste_1 ] [ ..... , [ liste_n ] ]
Zeiger = [ liste ]
```

für mehrdimensionales Feld
für eindimensionales Feld

Zeiger

instanziiertes Objekt der Objektklasse Array, also Ableitung vom Script-
Objekt Array per `var Zeiger = new Array();`



[] bedeutet hier **nicht** optional !!

[liste_1] bis [liste_n]	Liste aus kommasetrennten Elementen Liste stellt ein symbolisches Feld da
--------------------------	--

liste	Liste aus kommasetrennten Elementen
<pre> 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 </pre>	

Listenelement: kann Ausdruck sein, der Wert liefern muss
kann entfallen, aber das trennende Komma muss kodiert werden
Folge der Listenelemente entspricht Initialisierungsfolge des Feldes
pro Komma eine automatische Indexerhöhung
wenn Element nicht kodiert, so wird das Feldelement
zum Index nicht initialisiert ----> Feldelement hat keinen

Datentyp

Hinweise: Feldindex beginnt ab 0
Anzahl der Feldelemente: 0 so Feld leer
sonst ab 1

Beispiele für ein instanziiertes und leeres Feld mit dem Bezeichner "Feld":

```
Feld = [1,2,3];      Element an Index 0 hat Wert 1
                     Element an Index 1 hat Wert 2
                     Element an Index 2 hat Wert 3
```

Anzahl der Elemente 3

Feld = [1,,,5];	Initialisiert wird nur mit Index 0	auf Wert 1
	mit Index 4	auf Wert 5
	Anzahl der Elemente 2	

Feld = [["Name", "Tom", "Tim", "Teo"], ["Alter", 6, 5, 4]];
Ein Feld, das aus genau 1 numerischen Wert besteht
darf nicht wie folgt deklariert werden:

```
var t new Array(33); // 33 als Wert ist falsch, sondern hier als Anzahl der Feldelemente
```

```
var t new Array()  
t[0]=33;           // ist korrekt
```

```
var t=new Array();
```

alert(t) zeigt nichts an
alert(!= null) zeigt true an

Beispiele für lokaler Feldzeiger

```

var X_Feld=new Array();
...

function test();
{var X00=X_Feld;           // erzeugt Zeiger als KOPIE
  var X01=X00[0];          // lesen aus Kopie
  var X01=12;              // Schreiben in Kopie und NICHT in X_Feld
X_Feld[0]=0;              // Schreiben in Feld und NICHT in Kopie
}
```

Beispiele für Mehrfachindex eines Feldes

```
var X_Feld=new Array();
X_Feld[0]=new Array();           // Feldelement 0 ist selbst Feld (Unterfeld)
X_Feld[0][0]=10;                 // Feldelement 0 als Unterfeld: Unterfeld-Element 0 belegen
```

Beispiele für null-Zuweisung

Variable deklariert ohne Wert: Wert ist undefiniert
Zuweisung dieser Variablen auf eine andere: Diese andere wird ebenfalls "undefined".

```
var t;           // undefined
var s=new Array();
```



```
s[0]=1;           // mit Wert
s[0]=;           // undefined --> ES WIRD null-Zeiger gebildet so dass s[0] != null false ergibt
                // das Feldelement bleibt aber erhalten, so dass .length nicht verändert wird
```

3.3.2. boolean Datentyp (Basis-Datentyp)

basiert auf dem Script-Objekt Boolean

hat Werte mit ausschließlich folgender Schreibweise **true**
false

also nicht 0
nicht 1
nicht TRUE
nicht True
nicht "true" etc. (siehe Script-Objekt Boolean)

Beispiel: if (10 > 20)

3.3.3. date Datentyp (Basis-Datenstruktur)

basiert auf dem Script-Objekt Date

siehe dort

3.3.4. function Datentyp (Basis-Datenstruktur)

basiert auf dem Script-Objekt Function

siehe Script-Objekt und Funktion

3.3.5. Literal Datentyp (Basis-Datentyp)

Wertkonstante, die **keiner** Variablendeklaration bedarf

Literal immer in ' ' bzw. " " kodieren:

\'	Entwertung des Zeichen '	z.B. \"	bedeutet das Literal '
\"	Entwertung des Zeichen "	z.B. "\"	bedeutet das Literal "

" bzw. "" als leeres Literal möglich

Zeichen kann 16-Bit-Unicode sein (also auch Zeichensatz ab 256)

ASCII: pro Zeichen 7 Bit verwendet, also 128 Zeichen möglich (0-127)
enthalten im Unicode

Unicode: pro Zeichen 16 Bit verwendet, also 65535 Zeichen möglich
Zeichen 0-127 identisch mit ASCII-Code

Kodierung als ESC-Sequenz \uxxxx mit xxxx als Hexaziffern

Beispiele:	\u0008	Rückschritt
	\u0009	horizontaler Tabulator
	\u000A	Zeilenvorschub (Line Feed) neue Zeile
	\u000B	vertikaler Tabulator
	\u000C	Seitenvorschub (Form Feed)
	\u000D	für Wagenrücklauf
	\u0020	Blank, Leerzeichen
	\u0022	"
	\u0027	'
	\u005C	\

Escape Sequenzen werden immer in Literal konvertiert:

\b	Backspace, Rückschritt
\f	Form feed, Blattvorschub
\n	Line feed (newline), Zeilenvorschub
\r	Carriage return , Wagenrücklauf (nicht Enter !!)
\t	Horizontal tab (Ctrl-I), horizontaler Tabulator
\'	Entwertung des Zeichen '
\"	Entwertung des Zeichen "
\\	Backslash z.B. für lokale Pfadangabe wie C:\\test\\bilder\\test.gif.
\\\\	Entwertung Backslash
\xhh	hexadezimal aber nur von 00 bis FF
\uhhhh	Unicode

Beispiele:	\u0008	Rückschritt
	\u0009	horizontaler Tabulator
	\u000A	Zeilenvorschub (Line Feed) neue Zeile
	\u000B	vertikaler Tabulator
	\u000C	Seitenvorschub (Form Feed)
	\u000D	für Wagenrücklauf



\u0020	Blank, Leerzeichen
\u0022	"
\u0027	'
\u005C	\

Ganzzahl: dezimal, hexa, oktal

Bsp.:	'18'	dezimal
	'0x12'	hexa
	'022'	oktal

ab Javascript 1.5 im Netscape 6.x ist oktal deprecated

Fließkomma: IMMER mit Dezimalpunkt

Bsp.:	'12e34'
	'5E-5.'

boolean: 'true' oder 'false'

null ist der null-Zeiger also nicht 0

3.3.6. number Datentyp (Basis-Datentyp)

basiert auf dem Script-Objekt Number

Integer:	Ganzzahl Zahlenbasis	immer 32 Bit breit dezimal oktal hexadezimal
	oktal	nur Ziffern 0 bis 7 erste Ziffer muss 0 sein auch negativ
	hexadezimal	Ziffern 0 bis 9 sowie Buchstaben A bis F (Gross-klein egal) ersten zwei Zeichen müssen sein 0X oder 0x auch negativ

Floating point: Gleitkommazahl, immer 64 Bit breit
Exponential-Darstellung möglich
Tausender-Trenner ist Komma
Dezimalkomma ist Punkt

NaN Not a Number
Wert wird geliefert bei numerischer Operation mit nicht-numerischen Wert z.B. Zeichenkette

Infinity Wert wird geliefert bei numerischer Operation, dessen Wertebereich den von Script überschreitet

-Infinity Wert wird geliefert bei numerischer Operation, dessen Wertebereich den von Script überschreitet

Beispiele:

oktal	0377
hexadezimal	0x37CF 0x37cF 0x37Cf 0x37cf 0X37CF 0X37cF 0X37Cf 0X37cf
Floating point	0.0001 kann auch kodiert werden als 00.0001 .0001, 1e-4, 1.0e-4 3.45e2

3.3.7. string Datentyp (Basis-Datenstruktur)

basiert auf den Script-Objekt String

Zeichen kann 16-Bit-Unicode sein (also auch Zeichensatz ab 256)

ASCII: pro Zeichen 7 Bit verwendet, also 128 Zeichen möglich (0-127)



enthalten im Unicode
 Unicode: pro Zeichen 16 Bit verwendet, also 65535 Zeichen möglich
 Zeichen 0-127 identisch mit ASCII-Code

Kettenlänge nur begrenzt aufgrund Vorgaben des Betriebssystems/Browsers
 Bsp.: Netscape-Browser hat max. 256 Zeichen pro Zeile

Eine Zeichenkette kann im Quellcode **nicht** über mehrere Zeilen gehen !
 Es sind die Methoden des Stringobjektes oder der Stringoperator + zu verwenden.

Leerkette ist "" oder " (Zeichen " bzw. ' zweimal kodiert)

Escape Sequenzen nicht zulässig, sondern immer als Literal kodieren per Verkettungs-Operation

\b	Backspace, Rückschritt	
\f	Form feed, Blattvorschub	
\n	Line feed (newline), Zeilenvorschub	
\r	Carriage return , Wagenrücklauf (nicht Enter !!)	
\t	Horizontal tab (Ctrl-I), horizontaler Tabulator	
\'	Entwertung des Zeichen '	
\"	Entwertung des Zeichen "	
\\	Backslash z.B. für lokale Pfadangabe wie C:\\test\\bilder\\test.gif.	
\\\\	Entwertung Backslash	
\\xhh	hexadezimal aber nur von 00 bis FF	
\\uhhhh	Unicode	
Beispiele:	\\u0008	Rückschritt
	\\u0009	horizontaler Tabulator
	\\u000A	Zeilenvorschub (Line Feed) neue Zeile
	\\u000B	vertikaler Tabulator
	\\u000C	Seitenvorschub (Form Feed)
	\\u000D	für Wagenrücklauf
	\\u0020	Blank, Leerzeichen
	\\u0022	"
	\\u0027	'
	\\u005C	\\

3.3.8. Objekttyp oder Objektklasse (Basis-Datenstruktur)

vordefiniert per Script-Objekt Object
 vom Programmierer frei definiert als Konstruktor zum Erzeugen eines Objektes per new Anweisung:
 Konstruktor ist der Bezeichner z.B. des Script-Objektes (auch Script-Objekt Objekt)

3.3.9. Zeigertyp (Basis-Datentyp)

Zeigerverwendung

Sämtliche Objekte und Instanzen werden über Zeiger referenziert. Selbst ein Literal, das Ergebnis eines Ausdrucks, ein Funktionsargument oder eine Methode bzw. Eigenschaft eines Objektes erhalten je einen Zeiger.

Nur über Zeiger sind auch Adressbereiche im Hauptspeicher ansprechbar, wobei die Zeiger natürlich in Zeiger laut Speicherverwaltung zum Betriebssystem automatisch umgewandelt werden.

Beispiel für Verwaltung der Zeiger bei dynamischen HTML-Objekten:

Dynamische HTML-Objekte werden zur Laufzeit gebildet. Dazu muss die Methode document.write() verwendet werden. Des weiteren ist es möglich, bei mehreren Objekten gleicher Art ein Feld als Sammlung aller Zeiger zu verwenden.

```
// internes Zeigerfeld für Objekte
var ObjektZeigerFeld = new Array();           // dynamisches Feld

function ObjektErzeugen(ObjektNummer)
// Die Objektnummer muss >=0 sein
// eine lückenlose Vergabe ist nicht nötig.
{
    //      Objekt erzeugen, wobei der Name unbedingt mit der Objektnummer versehen werden muss
    document.write(      '<OBJECT ID="OBJECT_ID' + ObjektNummer.toString() + '" ></OBJECT>');

    //      und dessen Zeiger merken in ObjektZeigerFeld[]
    document.write(      '<SCRIPT LANGUAGE="JavaScript1.2">'
        + 'ObjektZeigerFeld[' + ObjektNummer
        + ']=document.OBJECT_ID' + Index.toString() + ';'
        + '</SCRIPT>'
    );
}
```



Zeigererzeugung im Browser (interner Zeiger)

Der Browser erzeugt Zeiger nicht immer automatisch:

Z.B. ist ein Funktionsargument im Kopf der Funktion ein Platzhalter für den Parameter laut Funktionsaufruf.

Aber **ohne** kodierten Platzhalter weiß der Browser nicht, wie er den Parameter zuordnen soll, weil kein interner Zeiger gebildet werden konnte.

Zeiger, die vom Browser erzeugt werden, stehen immer im Kontext zum per Zeiger referenzierten Objekt bzw. zur Instanz:

Die Gültigkeit der Zeiger ist die der Gültigkeit des Objektes bzw. der Instanz.

Es besteht die Möglichkeit, über Script auf vom Browser erzeugte (interne) Zeiger zuzugreifen.

Zeigerzuweisung**Beispiel 1**

```
<HTML>
<HEAD>
<script>
  function test(X0)
  // X0 Zeiger auf ein Objekt
  {
    var X1=X0.style;
    var X2='color';
    var X3;                // Zeiger auf X0.style.color
    var X4;                // Nullzeiger
    var X5='white';

    alert('Eigenschaft X0.style.' + X2 + ' vorhanden ? ' + (X0.style.color !=null)); // true
    alert('Eigenschaft X1.' + X2 + ' vorhanden ? ' + (X1.color !=null));           // true

    X1.color='black';
    alert('Colorwert == black ? ' + (X1.color=='black'));

    eval('X3=X1.' + X2 + '); // liefert Zeiger von X0.style.color
    alert('eval von X3 erfolgt, also X3==X0.style.color ? ' + (X3==X0.style.color)); // true

    if (X3!=null)
    {
      X4=X3;
      alert('Zuweisung X4=X3 erfolgt, also X4==X0.style.color ? ' + (X4==X0.style.color)); // true
      alert('Zuweisung X4=X3 erfolgt, also X4==X1.color ? ' + (X4==X1.color));           // true
      alert('X4 == black ? ' + (X4 == 'black'));                                         // true
    }

    // ABER:
    //      Wird ein Zeiger mit einem NICHT-Zeiger überschrieben, so wird der Zeiger ebenfalls zum Nicht-Zeiger.
    //      Grund: Javascript passt das Ziel der Quelle an auch mit Typwechsel der Zeiger-Variablen
    //      von Zeiger auf Nicht-Zeiger
    //      X4=X5; ist eben NICHT das Belegen von X0.style.color SOMDERN die Umwandlung vom Zeiger X4 in
    //      einen
    //      String !!!!
  }
</script>
</HEAD>
<BODY>
<DIV ID="TEST">
  Test-DIV
</DIV>
<script>
  test(TEST);
</script>
</BODY>
```

Beispiel 2

Es gibt diverse Ausnahmen: Objekt wie z.B. Style.

Da das Style-Element z.B. visibility über einen Zeiger verfügen muss, damit es einen Wert ablegen kann ab der Adresse laut Zeiger,



darf die Wertzuweisung nicht den Zeiger löschen ! Dafür sorgt das Objekt selbst.

```
function test(X00,X01,X02)
// X00 Zeiger auf Feld, das Objektzeiger enthält
// X01 Style-Eigenschaft z.B. 'visibility', kein ':' kodieren, darf NICHT Leerkette sein
// X02 Stylewert der Eigenschaft laut X01
//      muss syntaxgerecht sein
//      Bsp.: X01 ist 'visibility'
//      X02 ist 'hidden'
//      wenn Leerkette so Eigenschaft im Wert gelöscht:
//      Da Eigenschaften öfters einen Standardwert haben, muss getestet werden,
//      was mit dem Löschen passiert, also ob Standardwert wieder aktiv wird !
//      kein ':' kodieren
{
    var X03=0;           // Länge
    var X04=0;           // Zähler
    var X05;             // Zeiger auf Style

    // Länge von Feld der Objektzeiger
    X03=X00.length;
    if (X03>0)
    {
        // prüfen ob Style-Eigenschaft nicht leer ist
        if (X01 !=")
        {
            // Felder der Objektzeiger abklappern
            for (X04=0; X04 < X03; X04++)
            {
                // Stylezeiger holen
                X05=X00[X04].style;
                // style ist nicht komplett belegbar mit Stringwerten:
                //      wenn X05=X00[X04].style dann bewirkt X05=X02; die
                //      Umwandlung von X05 ist String wie X02
                //      und nicht X00[04].style=X2
                //      wenn X05=X00[X04] dann bewirkt X05.style=X02; die
                //      Umwandlung der Eigenschaft Style
                //      von Zeiger nach String wie X02
                //      o dass alert(X05.style); einen Scriptfehler bringt,
                //
                //      tyle nicht existent
                //      (Mitglied nicht gefunden: style muss Zeiger sein,
                //      kein String)
                // Hinweis: style ist nur durch style eines vorhandenen Objektes ersetzbar, also
                // durch kopieren der style-Zeiger, wobei das vorhandene Objekt
                // NICHT gelöscht werden darf: Vorhandenes Objekt als
                //
                //      unsichtbares Vorlageobjekt ist möglich, da trotz Zeigeridentität
                //
                //      style sich das ID vom Vorlageobjekt und das ID vom Objekt mit
                //      identischem style-Zeiger unterscheiden
                // Zeiger der Style-Eigenschaft holen und Wert zu weisen:
                //      Das Style-Objekt sorgt dafür, dass mit Zuweisung des Wertes
                //      NICHT die Style-Eigenschaft von Zeiger zu String wird, sondern
                //
                //      String als Wert ab Adresse laut Zeiger abgelegt wird.
                //      Ein alert auf die Style-Eigenschaft zeigt NICHT [object] an
                //
                //      immer den Wert der Style-Eigenschaft.
                eval('X05.' + X01 + '=' + X02 + ';');
                // Variablen, deren Inhalte kein String sind, müssen als 'variablen_bezeichner'
                //      kodiert werden.
                // Variablen, deren Inhalt String sind, dürfen NICHT als 'variablen_bezeichner'
                //      kodiert werden, sondern nur als variablen_bezeichner.
            }
        }
    }
}
```

Zeigerkopie



Jeder Zeiger kann in eine Zeigervariable kopiert werden. Ist er gespeichert, so hat die Zeigervariable nicht unbedingt mehr mit dem Kontext des Objektes bzw. der Instanz zu tun. Es ist Sache der Programmierers, darauf zu achten.

Zeiger in HTML

Der Programmierer kann in HTML zu einem Tag durch die Kodierung der Attribute ID und/oder NAME einen Zeiger als Referenz auf diesen Tag erzeugen. Diese Referenz ist direkt in Script als Objektbezeichner per Punktnotation verwendbar.

Zeiger und Browserperformance

Die Browser-Performance zur Darstellung des HTML-Dokumentes oder eines Objektes kann wie folgt erhöht werden:

In HTML: Wenn das ID-Attribut zum Tag zulässig ist, dann immer kodieren.
 Wenn das NAME-Attribut zum Tag zulässig ist, dann immer kodieren.
 Wenn NAME- und ID-Attribut zum Tag zulässig sind, dann ID-Attribut kodieren,
 es sei denn, es wird eine Referenz über das NAME-Attribut benötigt.

In Script: Bei Referenz auf eine Methode oder Eigenschaften eines Objektes sollte immer **zuerst** ein Zeiger auf das Objekt erzeugt werden, um **dann** mit diesem Zeiger die Methode bzw. Eigenschaft anzusprechen. Das gilt **vor allem** für Objekte, die Kinder haben, welche aufgerufen werden sollen. Die Zeigerbildung ist natürlich nur dann sinnvoll, wenn das betroffene Objekt mehrmals im Scriptcode referenziert werden soll. Die Zeigerbildung erspart dem Browser die mehrmalige Berechnung des Zeigers. Je **länger** die Punktnotation ist, um so mehr Aufwand hat der Browser beim Parsen, denn für jede Ebene in der Notation muss der Zeiger der Ebene berechnet werden. Die Zeigerbildung schafft auch Übersichtlichkeit im Quellcode.

```
Bsp.: var BodyObjekt = document.body;
      BodyObjekt.eigenschaft = ....;
      und nicht document.body.eigenschaft = ....;

      var FeldDerZeigerImDokument = document.all;
      FeldDerZeigerImDokument.objekt_bezeichner.eigenschaft = ....;
```

Null-Zeiger

Das Literal **null** ist der null-Zeiger, also nicht numerisch 0 und nicht das Zeichen "0".

typisierter Zeiger

Dieser Zeiger ist natürlich ebenfalls eine Referenz. Aber der Zeiger **muss** auf eine Größe zeigen, die einen **bestimmten** Datentyp hat. Typisches Beispiel ist der Aufruf einer Funktion:

Der Aufruf einer Funktion darf nur dort stattfinden, wo eine Referenz zulässig ist, da der Funktionsbezeichner einen Zeiger repräsentiert. z.B. ist ein Funktionsaufruf innerhalb eines Literals nicht möglich.

Falls die Funktion etwas liefert, gilt zusätzlich: Anstelle des Funktionsaufrufes wird die gelieferte Größe gesetzt. Damit **muss** die Funktion auch **datentyp-gerecht** liefern (entspricht einem typisierten Zeiger).

Analog gilt das auch für den Aufruf einer Methode.

Typischer Aufruf erfolgt z.B. innerhalb eines Ausdrucks:

Beispiel:

```
function Test(Wert1, Wert2)
{return ( Wert1 + Wert 2);}

var Wert = 20 + Test(10,20);

alert(Wert.toString());

// Nachfolgender Aufruf ergibt einen Fehler, da die Funktion einen numerischen Wert liefert
// Die Methode alert() kann eventuell automatisch nach String konvertieren.
alert( "Das Ergebnis lautet " + Test(10,20));
```

Zeiger auf den Konstruktor

Die Eigenschaft `.constructor` liefert den Bezeichner einer JScript-Objektklasse (Objekttyp) oder eines privaten Konstruktors.

Anwendung: Ermittlung der Objektklasse/Konstruktors eines abgeleiteten Objektes.

Ableitung aus der Objektklasse

per Anweisung `new` mit der Objektklasse als Konstruktor benötigt (siehe dort)



nicht bei Script-Objekt Math möglich

Ableitung aus privatem Konstruktor
per Anweisung new, die den privaten Konstruktor verwendet

Syntax:

```
[ var Wert = ] object.constructor
```

Wert Bezeichner, der nicht innerhalb " " bzw. ' ' liegt

JScript-Objektklassen sind z.B.

	Array
Boolean	
Date	
Function	
Number	
String	

Bezeichner eines privaten Konstruktors

object Zeiger auf ein per abgeleitetes Objekt

Beispiel 1 für Ableitung aus einem JScript-Objekt:

```
var Kette = new String("Hi");
alert( (Kette.constructor == String));      // liefert "true"
alert( (Kette.constructor == "String"));      // liefert "false"
```

Beispiel 2 für Ableitung anhand privaten Konstruktors:

```
function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion(); // bewirkt Ausführung von TestFunktion() also auch von alert()

// ZeigerAufFunktion();      // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
//                              // da keine Ableitung vom JScript-Objekt Function
// Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration erkannt

alert(ZeigerAufFunktion.constructor == TestFunktion);      // true
alert(TestFunktion.constructor == TestFunktion);      // false
```

3.3.10. frei definierbarer Datentyp anhand Script-Objekt Object

Das Script-Objekt Object, stellt sozusagen den Prototyp aller anderen Objekte dar. Das Script-Objekt Object hat im Gegensatz zu anderen Script-Objekten wie Array oder Date keinen speziellen Datentyp implementiert. Die new Anweisung liefert also einen untypisierten Zeiger. Das Script-Objekt Object ist also für den Programmierer ein **symbolisches** Objekt, das als Objektklasse (Konstruktor) per new Anweisung für die Erzeugung eines einfachen Objektes benutzt werden kann, um es per Prototyping mit gewünschten Eigenschaften und Methoden zu erweitern und damit Datentypen beliebiger Art zu implementieren. Das Objekt ist also ideal für die Erzeugung eigener und freier Datenstrukturen anhand der Basis-Datentypen und Basis-Datenstrukturen.

3.4. Variable

Komponenten der Variable

Eine Variable besteht aus den Komponenten

Namen (Bezeichner) der Variablen, den der Programmierer z.T. selbst festlegen kann,
und Inhalt (Wert) der Variablen, den der Programmierer z.T. selbst festlegen kann.

Der Name der Variablen ist der **Platzhalter** für den Zeiger, der erst zur Laufzeit gebildet wird und der dann auf die Instanz der Variablen im Speicher zeigt. Anhand des Variablennamens kann der Programmierer auf die Instanz zugreifen und z.B. der Variablen einen Wert zuweisen.

Vor dem Variablenzugriff (außer erste Wertzuweisung) muss die Variable initialisiert sein, damit der Datentyp feststeht.

Variable deklariert ohne Wert: Wert ist undefiniert
Browser meldet undefined, wenn null-Zeiger erkannt.

Zuweisung dieser Variablen auf eine andere: diese andere wird ebenfalls "undefined".

Beispiel: var t; // undefined
 var s=new Array();



```

s[0]=1;           // mit Wert
s[0]=t;           // undefined --> ES WIRD null-Zeiger gebildet so dass s[0] !=
null
//               false ergibt
//               // das Feldelement bleibt aber erhalten, so dass .length nicht
//               verändert wird

```

Globale Variablen durch Funktion verändern

Zuweisung des null-Zeigers hebt nicht die var-Deklaration auf !

```

<HTML><HEAD>
<SCRIPT LANGUAGE="JScript">
var d2760=1;           // global

function test()
{alert(d2760);         // 1
d2760=null;
alert(d2760);         // null
                        // Zuweisung des null-Zeigers hebt nicht die var-Deklaration auf !

d2760=0;
alert(d2760);         // 0
}

</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JScript">
test();                // 1 null 0
alert(d2760);         // 0
</SCRIPT>
</BODY>
</HTML>

```

Datentyp einer Variable

Ein Datentyp einer Variablen kann bei der Deklaration der Variablen **nicht kodiert** werden. Javascript ermittelt den Datentyp automatisch mit der ersten Wertzuweisung auf die Variable. Ist der Datentyp einmal festgelegt, so ist er nicht mehr änderbar. Datentyp-Konvertierung möglich.

Deklaration einer Variable

Es sollte zur Deklaration immer var kodiert werden (var-Anweisung).

Eine ohne var deklarierte Variable ist immer global !

Beispiel: Eine ohne var deklarierte Variable innerhalb einer Funktion ist
nicht funktionslokal
sondern global !

Eine fehlende var-Anweisung bewirkt die Überschreibung einer bereits

als global deklarierten Variablen

durch eine Variable und deren Wert, die mit gleichem Namen ohne var-Anweisung
später deklariert wird.

Ab Javascript 1.2

ist es Pflicht, die var-Anweisung zu verwenden

sollten eine Variablendeklaration **ihre eigene** Quelltextzeile besitzen und **nicht** mehrere Variablendeklationen in einer gemeinsamen Zeile kodiert sein (Vermeidung der kommagetrennter Deklaration von Variablen gleichen Typs)

Deklaration ansonsten sonst je nach Variablenart, wobei eine Initialisierung nicht erfolgen muss:

Eine deklarierte aber nicht initialisierte Variable hat den Wert undefined und den Zeigerwert null.

Bsp.:

```

if (Wert == null)      // if liefert true
var Wert;              // Variable deklariert aber ohne Wert, also Wert undefined automatisch zugewiesen
if (Wert == undefined) // if liefert true
if (Wert == null)      // if liefert true
var Wert=10;           // Variable deklariert aber mit Wert
if (Wert == undefined) // if liefert false
if (Wert == null)      // if liefert false

```

Vor dem Variablenzugriff (außer erste Wertzuweisung) muss die Variable initialisiert sein, damit der Datentyp feststeht.

var-Anweisung basiert auf dem gleichnamigen Javascript-Objekt "var"



Globale Variablen durch Funktion verändern

Zuweisung des null-Zeigers hebt nicht die var-Deklaration auf !

```
<HTML><HEAD>
<SCRIPT LANGUAGE="JScript">
var d2760=1;           // global

function test()
{alert(d2760);         // 1
 d2760=null;
 alert(d2760);         // null
                        // Zuweisung des null-Zeigers hebt nicht die var-Deklaration auf !
 d2760=0;
 alert(d2760);         // 0
}

</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JScript">
test();                // 1 null 0
alert(d2760);          // 0
</SCRIPT>
</BODY>
</HTML>
```

lokaler Feldzeiger

```
var X_Feld=new Array();
...

function test();
{var X00=X_Feld;        // erzeugt Zeiger als KOPIE
 var X01=X00[0];        // lesen aus Kopie
 var X01=12;             // Schreiben in Kopie und NICHT in X_Feld
 X_Feld[0]=0;            // Schreiben in Feld und NICHT in Kopie
}
```

Instanziierung einer Variablen

durch parsen der Deklaration der Variablen

es wird immer ein Zeiger erzeugt, denn die Variable liegt zur Laufzeit im Speicher und besitzt einen Zeiger:

Zeiger wird repräsentiert durch den Variablennamen (Variablenbezeichner)
 Zeiger kann auf die Abfrage der Variablenexistenz verwendet werden:

```
Bsp:   var variable=0;    // numerische Variable instanzieren
        if (variable) {...}
        oder   if (variable != null) {...}
```

Eine deklarierte aber **nicht** initialisierte Variable hat den Wert **undefined** und den Zeigerwert **null**.

```
Bsp.:
if (Wert == null)           // if liefert true
var Wert;                  // Variable deklariert aber ohne Wert, also Wert undefined automatisch zugewiesen
if (Wert == undefined)     // if liefert true
if (Wert == null)          // if liefert true
var Wert=10;               // Variable deklariert aber mit Wert
if (Wert == undefined)     // if liefert false
if (Wert == null)         // if liefert false
```

Vor dem Variablenzugriff (außer erste Wertzuweisung) muss die Variable initialisiert sein, damit der Datentyp feststeht.

Referenzierung einer Variable

immer über den Zeiger laut Instanziierung, also über den Variablennamen (Variablenbezeichner)

Eine deklarierte aber **nicht** initialisierte Variable hat den Wert **undefined** und den Zeigerwert **null**.

Bsp.:



```

if (Wert == null)           // if liefert true
var Wert;                  // Variable deklariert aber ohne Wert, also Wert undefined automatisch zugewiesen
if (Wert == undefined)     // if liefert true
if (Wert == null)         // if liefert true
var Wert=10;              // Variable deklariert aber mit Wert
if (Wert == undefined)    // if liefert false
if (Wert == null)        // if liefert false

```

Vor dem Variablenzugriff (außer erste Wertzuweisung) muss die Variable initialisiert sein, damit der Datentyp feststeht.

Löschung einer Variable

Die Zuweisung des Wertes **null**, also eines Null-Zeigers, bewirkt die Löschung der Variablen (inklusive Speicherplatz)

```

Bsp:    var variable=0;      // numerische Variable instanzieren
        variable=null;      // Zeiger löschen, also Instanz aufheben

```

Es kann auch die Anweisung delete verwendet werden, um das gesamte Objekt oder Teile davon zu löschen (inklusive Speicherplatz)

```

Bsp.:   var Wert = delete objekt_instanz.eigenschaft;

        Wert    true,    so Löschung erfolgreich
        false,   so Löschung nicht erfolgt

```

Script-Objekte sind nicht löschar

Nur per Prototyping zum Script-Objekt hinzugefügte Eigenschaften und Methoden sind löschar.

Gültigkeit einer Variable

immer laut Kontext der Kodierung zur Variablen: lokal oder global

Eine ohne var deklarierte Variable ist immer global !

Beispiel: Eine ohne var deklarierte Variable innerhalb einer Funktion ist nicht funktionslokal sondern global !

Eine fehlende var-Anweisung bewirkt die Überschreibung einer bereits als global deklarierten Variablen durch eine Variable und deren Wert, die mit gleichem Namen ohne var-Anweisung später deklariert wird.

lokal: wenn innerhalb einer Funktion **und mit** var-Anweisung deklariert

global: wenn auf der obersten Ebene im Quellcode
wenn nicht innerhalb einer Funktion
wenn **ohne** var-Anweisung deklariert

```

Beispiel: <SCRIPT>
        var DasIstEineGlobaleVariable = "Hallo";           // oberste Ebene im Quellcode
        var Kette = " und ich";

        function Test()
        {
            var DasIstEineLokaleVariable = ' Du';

            DasIstKeineLokaleVariableWeilVARfehlt = ' !';

            alert(
                DasIstEineGlobaleVariable
                + DasIstEineLokaleVariable
                + DasIstKeineLokaleVariableWeilVARfehlt
            );

            var Kette = " und wir";           // überschreibt nicht die globale Variable
                                           // trotz Namensgleichheit, sondern ist lokal
        }

        // nachfolgende Anweisung bringt Fehlermeldung
        alert(
            DasIstEineGlobaleVariable
            + DasIstEineLokaleVariable
            + DasIstKeineLokaleVariableWeilVARfehlt
        );

```



```
// nachfolgende Anweisung bringt keine Fehlermeldung
alert(      DasIstEineGlobaleVariable
      + DasIstKeineLokaleVariableWeilVARfehlt
    );

Kette = " und wir";           // überschreibt die globale Variable
</SCRIPT>
```

Fehlermeldungen

Es ist darauf zu achten, dass der Browser die HTML-Dokumente nicht aus dem Browser-Cache liest, solange die Dokumente nicht fehlerfrei laufen, also beim Test der Dokumente (Daten im Browser-Cache vor jedem Test löschen).

Das generelle Lesen aus dem Cache ist per Script, META-Tag, sowie per Browsereinstellung abänderbar. Letztere kann der User treffen, in dem er z.B. beim Internet Explorer die Cache-Löschung mit Schließen des Browser abhakt.

Es ist üblich, dass der User den Browser-Cache **nicht löscht**, da der Browser den Cache-Füllstand automatisch verwalten kann. Daher muss der Programmierer damit rechnen, dass die HTML-Dokumente aus dem Cache geladen werden könnten. Deshalb ist es wichtig, dass der Programmierer der Webseiten dafür sorgt, dass per META-Tag immer ein Datum der Webseite besteht, das mit Sicherheit verfallen ist und somit die HTML-Daten vom Server geladen werden. Achtung: Des erneute Laden einer Webseite (Reload), z.B. innerhalb eines bestimmten Zeitraumes, sollte aus dem Cache kommen, damit nicht Daten doppelt zu laden sind, wenn sie bereits gültig im Cache liegen. Will der Programmierer absolut sichergehen, dass nach einem bestimmten Zeitraum die Daten vom Server zu laden sind **und** soll der User dieses Verhalten nicht beeinflussen können, dann muss das Verhalten per Script programmiert werden (Algorithmus ist Sache des Programmierers). Zugleich wird der User und der Server mehr Traffic beim Laden der Daten der HTML-Dokumente haben.

Fehlermeldung beim Internet Explorer zu Objekten:

z.B. "Objekt nicht gefunden" oder "Objekt ... unterstützt Eigenschaft nicht" oder "CLASS nicht gefunden"

Die Fehler, die per Fehlermeldungen zu Objekten im Internet Explorer angezeigt werden, können dessen im Speicher instanzierte Objektstruktur durcheinander bringen, so dass nach Abhilfe des Fehlers im Quelltext und anschließender Aktualisierung des Dokumentes (per Aktualisierungs-Button im Browsermenü) trotzdem dieselbe Fehlermeldung erscheint. Es ist daher ratsam, bei solchen Fehlermeldungen den Browser zu schließen, den Browser-Cache zu löschen und dann den Browser mit dem Quelltext neu zu starten. Ursache ist ein eventuelles automatisches aber fehlerhaftes Prototyping von Objekten (auch von browserinternen). Hinweis: Man sollte immer mit genau 1 Instanz des Browsers testen, damit Objektstrukturen konsistent bleiben können.

Fehlermeldungen zum Script

z.B. "Scriptfehler"

Fehlermeldung kann echten Fehler anzeigen (z.B. wegen falscher Browserversion) **oder** Speichermangel, welcher die Abarbeitung des Skriptes verhindert (nicht mögliche Instanzierung von Objekten, Variablen etc.). Bei Speichermangel kann der Neustart des Browsers helfen.

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

Bsp.: var Kette = 'test'; + '.jpg'; // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen +
 alert(Kette); // zeigt nur test an und **nicht** test.jpg

Bsp.: for (var i = 0; i <3; i++); // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen {
 {alert(i);} // genau eine Anzeige, die 3 anzeigt

3.4.1. nicht Objektvariable (Variable nicht per new erzeugt)

var bezeichner_liste [= wert_oder_ausdruck];

var bezeichner = (bedingung) ? wert_oder_audruck1 : wert_oder_audruck2;

```
//entspricht     if (bedingung)
//               { bezeichner = wert_oder_audruck1; }
//               else
//               { bezeichner = wert_oder_audruck2; }
//               liefert wert_oder_audruck1 wenn bedingung true liefert
//               wert_oder_audruck2 wenn bedingung false liefert
```

bezeichner_liste: Kommatrennung

Bezeichner: beginnt mit Buchstabe oder Unterstrich
 sonst Buchstabe, Ziffer, Unterstrich

wert_oder_ausdruck: optionale Variablen-Initialisierung

Wert für Zeichenketten-Variable ist ein Literal, also in " " oder ' ' kodieren



auch Kombination von Literalen, Variablen
 Variablentyp: definiert mit Typ des Wertes bzw. des Ausdrucksergebnisses

= Zuweisungsoperator

var var Anweisung

3.4.2. Objektvariable

Eine Objektvariable ist eine Variable, die anhand eines Objektes erzeugt wird und mit dem Namen der Variablen einen Zeiger auf ein Objekt, also eine Objektinstanz im Speicher repräsentiert.
 siehe auch Datentypen und Objektbeschreibungen

Objektklasse bzw. Objekttyp

Das Objekt, welches zur Deklaration der Objektvariablen verwendet wird, kann auch als **Objektklasse** oder **Objekttyp** bezeichnet werden. Die Objektklasse dient also zur Ermittlung des Typs der Objektvariablen. Z.B. können sind alle Javascript-Objekte als Objektklasse verwendet werden.

Der Programmierer kann unter Javascript eigene Objektklassen definieren. Das geht aber nur, in dem der Konstruktor der new-Anweisung eine Funktion darstellt. Diese Funktion erzeugt das komplette Objekt. In der Funktion können natürlich auch Objektvariablen auftauchen, die sich auf **andere** Objektklassen beziehen (auch private).

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
 Es wird die Eigenschaft .prototype **nicht** erzeugt !

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
    this.nummer = nummer;
    this.plz = plz;
    this.ort = ort;
    this.methode = methode;
}

person = new datenstruktur_erzeugen
(
    "Erika",           // Vorname
    "Mustermann",     // Nachname
    "Musterstrasse",  // Strasse
    "1",              // Nummer
    "10000",          // PLZ
    "Musterstadt",    // Ort
    datenstruktur_ausgeben // ohne () kodieren
);

// alternativ auch kodierbar:
// var person = {    vorname:"Erika",    // Vorname
```



```
//      nachname:"Mustermann",      // Nachname
//      strasse:"Musterstrasse",      // Strasse
//      nummer:"1",                  // Nummer
//      plz:"10000",                  // PLZ
//      ort:"Musterstadt"              // Ort
//      methode:datenstruktur_ausgeben// Ausgabemethode ohne () kodieren
//      };

// Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

alert(person.vorname + "\n" + person.methode);      // person.methode ohne () kodieren !
// -->
</SCRIPT>
</HTML>
```

vordefinierte und browserinterne Objekte

Javascript-Objekte und die Objekte des Browsers sind in der Scriptmaschine vordefiniert, also implementiert. Sie können als Objektklasse verwendet werden.

private Objekte

Der Programmierer kann **private Objekte** per new-Anweisung definieren. Die Scriptmaschine kann nur dann private Objekte verarbeiten, wenn diese auf vordefinierten Objekten basieren und/oder der Programmierer sämtliche Eigenschaften und Methoden zum Objekt programmiert, wobei diese Eigenschaften und Methoden letztendlich **immer** auf vordefinierten Objekten und vordefinierten Datentypen basieren müssen (siehe auch Datentypen und Objektbeschreibungen). Der Programmierer kann unter Javascript eigene Objektklassen definieren.

Für private Objekte, die nicht aus einem vordefinierten Objekt abgeleitet werden, wird leider nicht die Eigenschaft .prototype erzeugt und ist somit nicht verwendbar !

Beispiel für Erweiterung des Script-Objektes Array, das die Eigenschaft .prototype besitzt:

```
function MaximumErmittleIn( )
{
    var max = this[0];      // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                           // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmittleIn;      // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6);      // 6 numerische Elemente

// neue Methode des JScript-Objektes Array aufrufen
alert(Feld.NeueArrayMethode());
```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
Es wird die Eigenschaft .prototype **nicht** erzeugt !

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }
}
```



```

    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
    this.nummer = nummer;
    this.plz = plz;
    this.ort = ort;
    this.methode = methode;
}

person = new datenstruktur_erzeugen
(
    "Erika",                // Vorname
    "Mustermann",          // Nachname
    "Musterstrasse",        // Strasse
    "1",                   // Nummer
    "10000",               // PLZ
    "Musterstadt",         // Ort
    datenstruktur_ausgeben // ohne () kodieren
);

// alternativ auch kodierbar:
// var person = {
//     vorname:"Erika",        // Vorname
//     nachname:"Mustermann",  // Nachname
//     strasse:"Musterstrasse", // Strasse
//     nummer:"1",            // Nummer
//     plz:"10000",           // PLZ
//     ort:"Musterstadt"      // Ort
//     methode:datenstruktur_ausgeben // Ausgabemethode ohne () kodieren
// };

// Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !
// -->
</SCRIPT>
</HTML>

```

instanzierte Objekte

Objekte, die bereits instanziiert sind, können als Objektklasse verwendet werden. **Nicht alle** vordefinierten Objektklassen sind bereits instanziiert (siehe auch Datentypen und Objektbeschreibungen).

Deklaration Objektvariable (Objektvariable von einer Objektklasse ableiten)

Eine Objektvariable muss speziell deklariert werden, je nach dem, um welche Objektklasse es sich handelt **und** ob diese bereits instanziiert ist oder nicht.

Wenn die Objektklasse **nicht bereits** instanziiert ist, so muss die var-Anweisung **in Verbindung** mit der Anweisung new bzw. per Literal kodiert werden. Die Objektklasse gilt dabei als Konstruktor in der new-Anweisung.

Wenn die Objektklasse **bereits** instanziiert ist, so muss die var-Anweisung **ohne** new-Anweisung verwendet werden. Es reicht eine

Zeigerzuweisung analog zu einer Nicht-Objekt-Variablen aus.

Es sollte **immer** eine Zeigerzuweisung programmiert werden, um den Zeiger weiterzuverwenden:

Bsp: document.all.objekt.eigenschaft // document.all ist instanziiertes Browser-Objekt
 ersetzen durch var Zeiger = document.all.objekt.eigenschaft;

Der Vorteil: Der Browser muss genau einmal die Objekthierarchie des Dokumentes abfragen, also nur



während der Zeigerbelegung. Danach ist die Zeigerverwendung ohne weitere Abfrage der Objekthierarchie möglich, was die Performance des Browsers erhöht.

Erweiterung Objektvariable (Prototyping)

Eine instanzierte Objektvariable kann per Eigenschaft `.prototype` um neue Eigenschaften und Methoden erweitert werden, wenn die Objektklasse die Eigenschaft `.prototype` besitzt.

Für private Objekte, die per `new`-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft `.prototype` erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft `.prototype` .
Nur für Objekte, die aus vordefinierten Objekten abgeleitet werden, wird die Eigenschaft `.prototype` erzeugt.

Nachfolgende Beispiele zeigen, wie ein externes Objekt mit seinem Datum und 1 Methode dynamisch zur Laufzeit des HTML-Dokumentes verwaltet wird, wobei die Kapselung der Daten im Objekt erfolgt. Wichtig ist es, dass der Javascript-Code innerhalb `<HEAD> ..</HEAD>` und außerhalb einer Funktion kodiert sein muss, da der Code mit **Laden** des HTML-Dokumentes und **vor** der Abarbeitung von `<BODY ...</BODY>` interpretiert werden muss.

Beispiel für Erweiterung des Script-Objektes Array, das die Eigenschaft `.prototype` besitzt:

```
function MaximumErmitteln( )
{
    var max = this[0];    // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                        // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmitteln;    // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6);    // 6 numerische Elemente

// neue Methode des JScript-Objektes Array aufrufen
alert(Feld.NeueArrayMethode());
```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:

Es wird die Eigenschaft `.prototype` **nicht** erzeugt !

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
```




```

        this.strasse = strasse;
        this.nummer = nummer;
        this.plz = plz;
        this.ort = ort;
        this.methode = methode;
    }

    person = new datenstruktur_erzeugen
    (
        "Erika",           // Vorname
        "Mustermann",     // Nachname
        "Musterstrasse",   // Strasse
        "1",              // Nummer
        "10000",           // PLZ
        "Musterstadt",     // Ort
        datenstruktur_ausgeben // ohne () kodieren
    );

    // alternativ auch kodierbar:
    // var person = {
    //     vorname:"Erika",           // Vorname
    //     nachname:"Mustermann",     // Nachname
    //     strasse:"Musterstrasse",   // Strasse
    //     nummer:"1",              // Nummer
    //     plz:"10000",             // PLZ
    //     ort:"Musterstadt",        // Ort
    //     methode:datenstruktur_ausgeben // Ausgabemethode ohne () kodieren
    // };

    // Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

    alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !
    // -->
</SCRIPT>
</HTML>

```

Beispiel für dynamisches Zeigerfeld:

```

var RumpfCodeDerFunktion1 =
    + 'if (Wert1 != 0)\n'
    + '{alert("Wert1 ist " + Wert1);}\n'
    + 'else\n'
    + '{alert("Wert1 ist Null !");}\n';

// internes Zeigerfeld für Objekte
var ObjektZeigerFeld = new Array(); // dynamisches Feld
// Es wird die Eigenschaft .prototype erzeugt !

function ObjektErzeugen(ObjektNummer)
// Die Objektnummer muss >=0 sein
// eine lückenlose Vergabe ist nicht nötig.
{
    // externes Objekt einbinden, wobei der Name unbedingt mit der Objektnummer versehen wird
    document.write( '<OBJECT ID="OBJECT_ID' + ObjektNummer.toString() + '"></OBJECT>' );

    // Zeiger vom externen Objekt im ObjektZeigerFeld[] merken
    document.write( '<SCRIPT LANGUAGE="JavaScript1.2">'
        + 'ObjektZeigerFeld[' + ObjektNummer.toString()
        + ']=document.OBJECT_ID' + ObjektNummer.toString() + ';\n'
        + '</SCRIPT>'
    );

    // globale Kodierung der objekt-eigenen Methode mit einem objekt-internen Bezeichner, der
    // eindeutig die Objektzugehörigkeit anzeigt
    // die Funktion wird zugleich instanziiert und ist somit per Zeiger adressierbar
    document.write( '<SCRIPT LANGUAGE="JavaScript1.2">'
        + 'function OBJECT_ID' + ObjektNummer.toString() + '_Funktion1()\n'
        + 'RumpfCodeDerFunktion1\n'
        + '</SCRIPT>'
    );

    // Erweiterung des Objektes per Prototyping um die Funktion1 durch deren Zeiger
    // den Wert1, der zugleich instanziiert und initialisiert wird
    document.write( '<SCRIPT LANGUAGE="JavaScript1.2">'

```



```

+ 'ObjektZeigerFeld[' + ObjektNummer.toString()
+ '].prototype.Funktion1= OBJECT_ID' + ObjektNummer.toString() + '_Funktion1\n'

+ 'ObjektZeigerFeld[' + ObjektNummer.toString()
+ '].prototype.Wert1= 0; // Initialisierung\n'
+ '</SCRIPT>'

);

}

```

Beim Prototyping von Funktion1 nicht () kodieren, da ja ein Zeiger übergeben werden soll !

Der Zugriff auf die Daten und Methoden des Objektes erfolgt über ObjektZeigerFeld[ObjektNummer].

```

z.B.      ObjektZeigerFeld[ObjektNummer].Funktion1();
          ObjektZeigerFeld[ObjektNummer].Wert1=33;

```

Die objekteneigene Funktion1 kann natürlich auch Parameter haben.

Die Doppelbezeichnung ein und desselben Funktionscodes mit

```

      "Funktion1"
und   "OBJECT_IDx_Funktion1"           x ist die jeweilige Objektnummer

```

macht Sinn, sobald mehrere Instanzen des externen Objektes gebildet werden sollen, da somit alle Objekte den selben Funktionsbezeichner haben, aber intern namentlich verschiedene. **Und:** Die Objekte haben je ihre **eigene** Funktion, die von keinem anderen Objekt benutzt werden kann. Das Prinzip der Kapselung wurde somit erfüllt. Der Funktionsname ist für den Programmierer einheitlich. Durch den Bezug per

```

ObjektZeigerFeld[ObjektNummer].Funktion1
ObjektZeigerFeld[ObjektNummer].Wert1

```

wird die Kapselung aufrechterhalten.

Diese Methode kostet Ressourcen und bläht die HTML-Datei auf ! Daher ist es natürlich auch möglich, nur 1 Methode zu deklarieren und dann den Objekten den identischen Zeiger zuzuweisen. Wichtig ist dabei, dass dann die Objekte **nicht parallel** auf diese Funktion zugreifen, da die Funktion in keinem Objekt gekapselt ist !!!

Durch die Auslagerung des Code der Funktion1 in die Stringvariable

```
RumpfCodeDerFunktion1
```

wird Übersicht erzeugt. Man beachte, dass

```

" " und " " sich nicht inkorrekt mischen !,
die maximal mögliche Gesamtlänge des Strings beachtet werden muss, also eventuell mehrere Variablen zu
kodieren sind.

```

Durch die Auslagerung der Variable

```
RumpfCode DerFunktion1
```

in eine eigene JS-Datei und deren Einbindung in das HTML-Dokument, kann der Aufbau der Funktion1, also des Objektes, von außen gesteuert werden, ohne das HTML-Dokument editieren zu müssen.

Empfehlenswert ist es, per \n am Zeilenende einen Zeilenumbruch zu erzeugen, damit die jeweilige erzeugte Zeile mit Ausführung von document.write() nicht unnötig zu lang wird, da Interpreter von Browsern nicht unbedingt überlange Zeilen ausführen können.

3.4.2.1. **Objektvariable mit new deklarieren**

Die Deklaration mit Anweisung new setzt ein nicht instanziiertes Objekt voraus.

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype . Nur für Objekte, die aus vordefinierten Objekten abgeleitet werden, wird die Eigenschaft .prototype erzeugt.

Syntax:

```
var Zeiger = new konstruktor[ (init_wert) ]
```

konstruktor Bezeichner der Objektklasse

init_wert je nach Objektklasse



```
var Zeiger = new Function( [parameter_liste] funktions_rumpf)

parameter_liste:    Kommatrennung

funktions_rumpf:    Liste aus kommasetrennten Scriptanweisungen
jede Scriptanweisung in " " oder ' ' setzen
```

Beispiel 1:

```
var Zeiger1 = new Array("1", 2, 33+44);

var Zeiger2 = new Boolean();

var Zeiger3 = new window.document;    // Diese Anweisung bringt Fehler, da windows.document instanziert ist !!

var Zeiger4 = new Otto;                // Diese Anweisung bringt Fehler, da Otto zuvor nicht definiert wurde !!

var PrivatesObjekt1 = new String('Private Objekt1');

var PrivatesObjekt2 = new PrivatesObjekt1; // Diese Anweisung bringt Fehler, da PrivatesObjekt1 instanziert ist !!

var PrivatesObjekt3 = new String('Private Objekt3');

var Zeiger = new Function("return init_wert;")
```

Beispiel 2:

```
function erzeuge_zwei_dim_feld(anzahl_spalten, anzahl_zeilen)
{
    // Spaltenfeld erzeugen
    this.spalten_feld= new Array(anzahl_spalten);

    // Zeilenfeld pro Spalte erzeugen
    for (var spalte=0; spalte < anzahl_spalten; spalte++)
    { this[spalte] = new Array(anzahl_zeilen); } //Zeilen-Felder
}
var zwei_dim_tabelle= new erzeuge_zwei_dim_feld(10,7); // 10 Spalten zu je 7 Zeilen
....
zwei_dim_tabelle[10,7]=22; // Spalte 10: 7Zeile: mit 22 belegen
```

Beispiel 3 für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
Es wird die Eigenschaft .prototype **nicht** erzeugt !

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function datenstruktur_ausgeben()
    {
        with (document)
        {
            write(person.vorname + " " + person.nachname + "<BR>");
            write(person.strasse + " " + person.nummer + "<BR>");
            write(person.plz + " " + person.ort + "<BR>");
            //      Erika Mustermann
            //      Musterstrasse 1
            //      .....
        }

        for (i in person)
        {
            document.write(i + ": " + person[i] + "<BR>");
            //      vorname: Erika
            //      nachname: Mustermann
            //      .....
        }
    }

    function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
    {
        this.vorname = vorname;
        this.nachname = nachname;
        this.strasse = strasse;
        this.nummer = nummer;
        this.plz = plz;
        this.ort = ort;
```



```

        this.methode = methode;
    }

    person = new datenstruktur_erzeugen
        (
            "Erika",           // Vorname
            "Mustermann",     // Nachname
            "Musterstrasse",   // Strasse
            "1",               // Nummer
            "10000",           // PLZ
            "Musterstadt",     // Ort
            datenstruktur_ausgeben // ohne () kodieren
        );

    // alternativ auch kodierbar:
    // var person = {
    //     vorname:"Erika",       // Vorname
    //     nachname:"Mustermann", // Nachname
    //     strasse:"Musterstrasse", // Strasse
    //     nummer:"1",           // Nummer
    //     plz:"10000",          // PLZ
    //     ort:"Musterstadt"     // Ort
    //     methode:datenstruktur_ausgeben // Ausgabemethode ohne () kodieren
    // };

    // Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

    alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !

// -->
</SCRIPT>
</HTML>

```

3.4.2.2. Objektvariable als Array Objekt aus Literalen deklarieren

Syntax:

```
var Zeiger = {literal_liste}
```

literalliste: kommagetrennte Elemente
Element

mit Aufbau "kette1";"kette2"

Doppelpunkt muss kodiert werden

kette 1 Eigenschaft des Literal-Objektes
 nur Plain-Text

kette2 Wert der Eigenschaft des Literal-Objektes
 nur Plain-Text

Beispiel:

```
var Index = "";
var Menge = { "a" : "Athen", "b" : "Berlin", "c" : "Paris", "d" : "Kairo" };
var Kette = "";
```

```

SchleifenAnweisung :           // ein Label (Marke)
{
    for (Index in Menge)
    {
        Kette = "Hauptstadt von ";

        switch (Menge[Index])
        {
            case "Berlin":      {
                                   Kette += "Deutschland: " + Menge[Index];
                                   // nicht weiter auf Athen und Kairo prüfen
                                   break;
                               }

            case "Athen":       {
                                   Kette += "Griechenland: " + Menge[Index];
                                   // kein break, also noch auf Kairo prüfen
                               }

            case "Kairo":       {
                                   Kette += "Ägypten: " + Menge[Index]; }
        }
    }
}

```



```

        default: {
            // im Falle von Paris:
            //      Kairo wird nie angezeigt, da im Feld
            //      hinter Paris

            break SchleifenAnweisung;
        }

        {alert(Kette);} // nicht bei Paris abgearbeitet
    }
}

```

3.5. Operatoren

Prioritäten:

niedrigste	Komma
	Zuweisungen aller Art
	? :
	&&
	^
	&
	== !=
	< <= > >=
	<< >> >>>
	+ -
	* / %
	! Tilde ++ etc. typeof
höchste	() [] Punkt

Prioritäten durch Klammerung aufhebbar

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

Bsp.:

```

var Kette = 'test'; + '.jpg'; // kein Syntaxerror wegen dem Semikolon vor dem Zeichen +
alert(Kette);                // zeigt nur test an und nicht test.jpg

```

Bsp:

```

for (var i = 0; i <3; i++); // kein Syntaxerror wegen dem Semikolon vor dem Zeichen {
{alert(i);}               // genau 1 Anzeige, die 3 anzeigt

```

3.5.1. Operatoren logische

logische Operationen liefern immer true oder false

	Oder	--> true wenn mindestens 1 Operand true
&&	Und	--> true wenn alle Operanden true
!	not	

3.5.2. Operatoren arithmetische

+ - / *	+ für Addition UND Zeichenverkettung / ist NICHT die Ganzzahldivision
%	Modulo Bsp: 13%5 ergibt 3, denn 13/5 ganz ist 2 mit Rest 3 Modulo liefert den Rest immer als Gleitkomma !

Ganzzahldivision existiert nicht --> selbst programmieren

Variante 1:

```

ergebnis = (zahl_1-(zahl_1 % zahl_2))
           / zahl_2;

```

Variante 2:

```

ergebnis= zahl_1 % zahl_2; // Gleitkommrest ermitteln
ergebnis= zahl_1 - ergebnis; // und von zahl_1 abziehen
ergebnis = ergebnis / zahl_2;

```

Variante 3: ergebnis = Math.floor(zahl1 / zahl2);

// liefert nächst kleinere ganze Zahl vom
// Gleitkomma-Ergebnis

ergebnis = Math.ceil(zahl1 / zahl2);

// liefert ganze Zahl vom Gleitkommaergebnis,
// die größer oder gleich dem Gleitkommaergebnis
// ist

Variablenoperatoren:

Variante 1:

bezeichner1+= bezeichner2; entspricht bezeichner1 = bezeichner 1 + bezeichner 2;



erst operieren dann zuweisen

Hinweis: auch für Strings möglich (Verkettung)

Wenn anstelle bezeichner2 mehrere Kettenvariablen

kodiert

werden, die mit dem Operator '+' verkettet sind, so diese Verkettungsfolge als Ganzes innerhalb eines runden Klammernpaares kodieren (also mit höherer Priorität als +=), damit vor Zuweisung der Verkettung auf bezeichner1 das Ergebnis der Verkettungsfolge als interner Zeiger vorliegt. Es könnte ansonsten dem bezeichner1 ein falscher Wert zugewiesen werden.

analog dazu für -= *= /= %=

Variante 2:

bezeichner++;

erst bezeichner verwenden, dann um 1 erhöhen

++bezeichner;

erst bezeichner um 1 erhöhen, dann verwenden

analog dazu für --

Hinweis: Math.ceil() dient zur Berechnung mit weiteren arithmetischen Operationsarten (siehe Math Objekt).

Dabein sind innerhalb () auch die Operatoren + - / * und % kodierbar.

3.5.3. Operatoren bitweise (Bitoperatoren)

bitweise Operationen (pro Bitposition)

liefern immer numerischen Wert

immer auf 32-Bit-Basis: für die Bitoperation werden Operanden auf 32-Bit-Breite umgewandelt

& neues Bit ist 1 wenn beide Operandenbits 1
 | neues Bit ist 1 wenn mindestens ein Operandenbit 1 ist
 ^ neues Bit ist 1 wenn beide Operandenbits verschieden sind
 ~ Tilde (auf Taste mit + und *) Negation

Bitverschiebungen: sind abhängig von der Scriptmaschine

nachfolgende Darstellung der Bitoperationen muss nicht zutreffen (ausprobieren)

bezeichner >> bit_anzahl

Bitverschiebung nach rechts:

reinkommende Bits sind 0

haben Wert des linken Bits

(an höchster Bitposition) VOR der Verschiebung

rausfallende Bits gehen verloren

bezeichner >>> bit_anzahl

Bitverschiebung nach rechts:

reinkommende Bits=0

rausfallende Bits gehen verloren

bezeichner << bit_anzahl

Bitverschiebung nach links:

reinkommende Bits=0

rausfallende Bits gehen verloren

bezeichner <<< bit_anzahl

gibt es leider **nicht**

3.5.4. Operatoren für Vergleich (Vergleichoperatoren)

Vergleichoperationen liefern immer true oder false

== gleich
 != ungleich

< > >= <=

in Teilmengenprüfung

Bsp. bezeichner1 in bezeichner2 --> true, wenn bezeichner1 Teilmenge von bezeichner2

Achtung: Werte können überhaupt nur gleich sein, wenn sie gemeinsamen Typs sind --> eventuell vorher konvertieren !

3.5.5. Operatoren für Verkettung von Zeichenketten

Bsp: 'Test' + '1' ergibt 'Test1'

3.5.6. Operator für Zuweisung

einfachste Form: variable_1 = wert

spezielle Form: Für variable_1 = variable_1 + variable_2; kann auch kodiert werden

variable_1 += variable_2;

Für variable_1 = variable_1 + wert_1;

kann auch kodiert werden

variable_1 += wert_1;



analog für - * / % << >> >>> & ^ |

3.5.7. Operator für Ermittlung des Datentyps (Operator typeof)

typeof operand;

Typen: "undefined"
"function"
"object"
"number"
"boolean"
"string"

Bsp: typeof 42 ergibt "number"

3.5.8. Ausdruck berechnen, aber den Wert nicht liefern (Operator void)

void(ausdruck) wertet den Ausdruck aus, aber liefert den Wert nicht

Bsp.: ... erzeugt Link ohne Verweiswert

3.5.9. this (Zeiger auf aktuelle Objekt-Instanz) (Anwendung auf Einzelanweisung zur Objektinstanz)

this.objekt

Beispiel 1 für Erzeugung eines 2-dimensionalen Feldes:

```
function erzeuge_zwei_dim_feld(anzahl_spalten, anzahl_zeilen)
{
    // Spaltenfeld erzeugen
    this.spalten_feld= new Array(anzahl_spalten);

    // Zeilenfeld pro Spalte erzeugen
    for ( var spalte=0; spalte < anzahl_spalten; spalte++)
    { this[spalte] = new Array(anzahl_zeilen); }
}

var zwei_dim_tabelle= new erzeuge_zwei_dim_feld(10,7); // 10 Spalten zu je 7 Zeilen
....
zwei_dim_tabelle[10,7]=22; // in Spalte 10 die Zeile7 mit 22 belegen
```

Beispiel 2 für HTML-Kodierung:

```
<INPUT
    TYPE=button
    NAME="logischer_button_name"
    onclick="alert(this.name)"
>
```

3.5.10. with (Zeiger auf aktuelle Objekt-Instanz) (Anwendung auf Block von Anweisung zur Objektinstanz)

with (objekt_instanz) { }

Alle Anweisungen innerhalb { } beziehen sich NUR auf die Instanz laut objekt_instanz.
Die Kodierung der Instanz per Punktnotation entfällt (vereinfachte Kodierung per with).

3.5.11. Operatoren in Microsoft JScript

Operatoren sind

+	Addition
-	Subtraktion
*	Multiplikation
/	Division (Gleitkomma)
%	Modulo
--	dekrementieren um 1
++	inkrementieren um 1
=	Zuweisung
&	bitweises AND
~	bitweises NOT
	bitweises OR



^	bitweises XOR
>>	bitweises Rechtsverschieben
<<	bitweise Linksverschieben
>>>	bitweises unsigned Rechtsverschieben
,	Trenner bei Aufzählung z.B. für Feld oder Funktions-Parameter-Liste
&&	logisches AND
!	logisches NOT
	logisches OR (zwei senkrechte Striche)
<	Wert-Vergleich auf kleiner
>	Wert-Vergleich auf größer
<=	Wert-Vergleich auf kleingleich
>=	Wert-Vergleich auf großgleich
==	Wert-Vergleich auf Identität (zwei Gleichheitszeichen)
!=	Wert-Vergleich auf Ungleichheit (bei String reicht die erst gefundene Ungleichheit)
===	Wert- und Datentyp-Vergleich auf Identität (drei Gleichheitszeichen)
!==	Wert- und Datentyp-Vergleich auf Ungleichheit (drei Gleichheitszeichen)
?:	if-Anweisung in anderer Form
-	Vorzeichen Minus
delete	Eigenschaft oder Methode von einem Objekt löschen (Achtung: nicht bei jedem Objekt zulässig)
instanceof	Element eines Feldes löschen (Achtung: nicht bei jedem Feld zulässig)
new	prüfen ob eine Instanz einer Objektklasse angehört
typeof	Erzeugung eines Objektes (Achtung: nicht bei jedem Objekt möglich)
void	Datentyp eines Ausdruckes
	Ausdruck liefert immer undefined bzw. null

Achtung: <<< Operator existiert leider **nicht** !

Hinweise zu den Vergleichsoperatoren

<	Vergleich auf kleiner
>	Vergleich auf größer
<=	Vergleich auf kleingleich
>=	Vergleich auf großgleich
==	Vergleich auf Identität (zwei Gleichheitszeichen)
!=	Vergleich auf Ungleichheit (erst reicht die erst gefundene Ungleichheit)
===	Vergleich auf Identität in Wert und Datentyp (drei Gleichheitszeichen)
!==	Vergleich auf Ungleichheit in Wert und/oder Datentyp (drei Gleichheitszeichen)

wenn ein Vergleichsoperand NaN, dann wird immer false geliefert

Stringvergleich in lexigraphischer Folge

-0 identisch mit +0

-Infinity	ist immer kleiner als alles andere numerische
Infinity	ist immer größer als alles andere numerische
NaN	ist immer ungleich zu allem anderen
null	bei Zeiger auf eine Instanz (Adresse der Instanz)
undefined	bei Werten (Inhalt einer Instanz)

Hinweise zum Operator %

Ganzzahl-Division und den Rest davon liefern

Syntax:

```
[ var Wert = ] number1 % number2;
```

number1 Integer oder Floating point
wenn Floating point, so automatisch auf Integer gerundet

number2 Integer oder Floating point
wenn Floating point, so automatisch auf Integer gerundet

Wert Integer

Ablauf: 1. Schritt interne_variable1 = number1 / number 2;

interne_variable1 automatisch erzeugt
wird auf nächsten kleineren Integerwert gesetzt



2.Schritt $\text{interne_variable2} = \text{interne_variable1} * \text{number2}$

interne_variable2 automatisch erzeugt

3.Schritt $\text{Wert} = \text{number1} - \text{interne_variable2}$

Beispiel: $33 \% 10$ $33 / 10 = 3,3$ in Periode, also 3

$3 * 10 = 30$

$33 - 30 = 3$

$19 \% 6.7$ $19 / 7 = 2,7...$ also 2

$2 * 7 = 14$

$19 - 14 = 5$

Hinweise zu den Operatoren

+	Addition bzw. Verkettung
-	Subtraktion
/	Division
*	Multiplikation
%	
<<	
>>	
>>>	
^	
=	Zuweisung

Es ist auch kodierbar

```
+=
-=
/=
*=
%=
<<=
>>=
>>>=
^=
|=
```

Beispiel: `var Wert = 10;`
`Wert +=23;`
`alert(Wert);` // zeigt 30 an

`Wert %=10;`
`alert(Wert);` // zeigt 3 an

Hinweise zum Operator <<

bitweise Linksverschiebung
 von rechts reinkommende Bits sind die Bits, die links rausfallen
 nach links rausfallende Bits kommen rechts wieder rein
 Syntax: `[Wert =] variable1 << variable2;`

`variable1` mit Wert der bitweise nach links zu verschieben ist

`variable2` numerisch
 Anzahl der Bits, um die verschoben wird

`Wert` mit Typ laut `variable1`

Beispiel:

`var Wert = -14 << 2`

mit -14 als Bitfolge

11111111 11111111 11111111 11110010

Wert nach Verschiebung in Bitfolge

111111 11111111 11111111 1111001011

Hinweise zum Operator >>



bitweise Rechtsverschiebung
 von links reinkommende Bits sind die Bits, die rechts rausfallen
 nach rechts rausfallende Bits kommen links wieder rein

Syntax: [Wert =] variable1 >> variable2;

variable1 mit Wert der bitweise nach rechts zu verschieben ist

variable2 numerisch
 Anzahl der Bits, um die verschoben wird

Wert mit Typ laut variable1

Beispiel:

var Wert = -14 >> 2

mit -14 als Bitfolge *11111111 11111111 11111111 11110010*

Wert nach Verschiebung in Bitfolge *1011111111 11111111 11111111 111100*

Hinweise zum Operator >>>

bitweise Rechtsverschiebung
 von links reinkommende Bits sind immer 0
 nach rechts rausfallende Bits bleiben weg

Syntax: [Wert =] variable1 >>> variable2;

variable1 mit Wert der bitweise nach rechts zu verschieben ist

variable2 numerisch
 Anzahl der Bits, um die verschoben wird

Wert mit Typ laut variable1

Beispiel:

var Wert = -14 >>> 2

mit -14 als Bitfolge *11111111 11111111 11111111 11110010*

Wert nach Verschiebung in Bitfolge *00111111 11111111 11111111 11111100*

Hinweis zum Operator ?:

siehe auch if-Anweisung

Syntax: ausdruck ? statement1 : statement2;

ausdruck: **muss** boolean liefern
 wenn true, so statement1 aktiviert
 wenn false, so statement2 aktiviert

statement Scriptanweisungen in " " kodieren

Hinweise zum Operator delete

Löschen einer Objekt-Eigenschaft oder Objekt-Methode oder des Objektes selbst
 bei Feld und Collection: Elemente **nur** durch objektteigene Methoden löschen
 JScript-Objekte sind nicht löschar:
 Nur per **Prototyping** zum JScript-Objekt hinzugefügte Eigenschaften und Methoden sind löschar

Syntax: [var Wert =] delete ausdruck;

ausdruck muss Zeiger liefern per
 Name einer Eigenschaft
 Referenz auf Feldelement

Wert true, so Löschung erfolgreich
 false, so Löschung nicht erfolgreich

Hinweise zum Operator instanceof



Syntax: [var Wert =] zeiger_auf_objekt **instanceof** objekt_klasse;

objekt_klasse ist ein Objekt, das im Browser automatisch implementiert wurde, aber nicht Teil des DOM des HTML-Dokumentes ist, also Objekte, die per new erzeugbar sind wie z.B. Date, Array, Object

Wert true, so Instanz existiert **und** Instanz wurde per Objektklasse instanziiert (z.B. per new)
false, so Instanz wurde nicht mit dieser Objektklasse instanziiert
oder Instanz existiert erst gar nicht

Beispiel:

```
function Testen(ZeigerAufObjekt)
{
    var IndexAlsString = "";
    var Kette = "";

    FeldDerObjektKlassen = new Array();

    // Feldelement mit Index "Date" füllen mit Instanz der Objektklasse Date
    FeldDerObjektKlassen["Date"] = Date;

    // Feldelement mit Index "Object" füllen mit Instanz der Objektklasse Object
    FeldDerObjektKlassen["Object"] = Object;

    // Feldelement mit Index "Array" füllen mit Instanz der Objektklasse Array
    FeldDerObjektKlassen["Array"] = Array;

    for (IndexAlsString in FeldDerObjektKlassen)
    {
        // Vergleich auf Klasse
        if (ZeigerAufObjekt instanceof FeldDerObjektKlassen[IndexAlsString])
        {
            Kette += "Objekt ist eine Instanz von " + IndexAlsString + "\n";
        }
        else
        {
            Kette += "Objekt ist keine Instanz von " + IndexAlsString + "\n";
        }
    }

    return(Kette);
}

var ZeigerAufDateObjekt = new Date();
alert (Testen(ZeigerAufDateObjekt));
```

Hinweise zum Operator typeof

Typ des Ergebnisses eines Ausdruckes liefern

Syntax:

[var Kette =] **typeof** ausdrück;
[var Kette =] **typeof**(ausdrück); // als Funktion

Kette	String
	"number"
	"string"
	"boolean"
	"object"
	"function"
	"undefined"

leider **nicht** "array" etc.

Hinweise zum Operator void

Ausdrucksergebnis immer auf **undefined** bzw. **null** setzen

Syntax:

void ausdrück



```

Beispiel:      if (void (10 + 30) == null )      // Klammern um 10 + 30, sonst gilt void 10, da Addition geringere
                                                    // Priorität als void hat
                { ..... }

```

Übersicht zu den Standard-Prioritäten der Operatoren für die Abarbeitung von Operationen:

höhere Priorität: der Operator wird zuerst verwendet
 von oben (höchster) nach unten (niedrigster)

<u>Operator</u>	<u>Funktion des Operators</u>
.	Punktnotation
[]	Feldnotation
()	Klammerung bzw. Funktionsparameterliste
++	Inkrementierung um 1
--	Dekrementierung um 1
-	Vorzeichen
~	bitweises NOT
!	logisches NOT
delete	
new	
typeof	
void	
*	Multiplikation
/	Division (Gleitkomma)
%	Modulo
+	Addition oder Verkettung
-	Subtraktion
<<	
>>	
>>>	
<	
<=	
>	
>=	
instanceof	
==	Vergleich auf Gleichheit
!=	Vergleich auf Ungleichheit
===	Vergleich auf Gleichheit (Wert und Datentyp)
!==	Vergleich auf Ungleichheit (Wert und Datentyp)
&	bitweises AND
^	bitweises XOR
	bitweise OR
&&	logisches AND
	logisches OR
?:	if-Abfrage
=	Zuweisung eines Wertes etc auf eine Instanz auch Kombination z.B. +=
,	Trenner in Aufzählung oder Liste

Aber: Innerhalb eines Ausdruckes gilt folgende Priorität von oben nach unten:

()	Klammerung
-	Subtraktion
+	Addition oder Verkettung
*	Multiplikation
/	Division
%	Modulo
=	Zuweisung

Fettgedrucktes ist die Abweichung von obiger Liste der Prioritäten aller Operatoren:

Die Priorität ist **vertauscht**. Diese Regelung in JScript klingt unsinnig, aber kommt dann zum Zuge, wenn Terme addiert **und** subtrahiert werden sollen: Es wird **zuerst** der Term für die Subtraktion berechnet, um **dann** das Ergebnis für die Addition mit einem anderen Term zu verwenden. Das interne Zwischenergebnis entsteht also nur in dieser Reihenfolge. Sollte also erst + und dann - kodiert sein, so bewirkt das Minus eine Klammerung. Empfehlung: Entweder alles klammern oder erst Minus und dann Plus kodieren.

Beispiel: var **Wert** = (10 * 20) + (100 - 10) + 30 - 20;

Schritt 1: 10 * 20 200 intern merken



Schritt 2:	100 - 10	90	intern merken
Schritt 3:	30 - 20	10	intern merken, wie Klammerung
Schritt 4:	200 + 90	290	
Schritt 5:	290 + 10	300	

nur im Ergebnis dasselbe wie

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	200 + 90	290	
Schritt 4:	290 + 30	320	
Schritt 5:	320 - 20	300	

Beispiel: var Wert = (10 * 20) + (100 - 10) - 20 + 30 ;

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	90 - 20	70	wie Klammerung
Schritt 4:	200 + 70	270	
Schritt 5:	270 + 30	300	

nur im Ergebnis dasselbe wie

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	200 + 90	290	
Schritt 4:	290 - 30	270	
Schritt 5:	270 - 30	300	

Beispiel: var Wert = (10 * 20) + 100 - 10 - 20 + 30 ;

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	90 - 20	70	intern merken
Schritt 4:	200 + 70	270	
Schritt 5:	270 + 30	300	

Beispiel: var Wert = (10 * 20) + 100 - 10 + 30 - 20;

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	30 - 20	10	intern merken
Schritt 4:	200 + 90	270	
Schritt 5:	290 + 10	300	

Beispiel: var Wert = -20 + (10 * 20) + (100 - 10) + 30 ;

Schritt 1:	10 * 20	200	intern merken
Schritt 2:	100 - 10	90	intern merken
Schritt 3:	-20 + 200	180	
Schritt 4:	180 + 90	270	
Schritt 5:	270 + 30	300	

Empfehlung: Um Prioritätenfolgen zu vermeiden oder zu ändern (oder falls man sich diese nicht merken kann, da andere Programmiersprachen wieder andere Regeln haben), sind Klammernungen sinnvoll.

Achtung: Jede Klammerung erzeugt eine interne Stack-Speicherung des Ergebnisses aus dem Ausdruck in der Klammer. Der Stack ist **nur endlich groß**.

Beispiel: var Wert = (10 * 20 + 100) - 10;

Schritt 1:	10 * 20	200
Schritt 2:	200 + 100	300
Schritt 3:	300 - 10	290

Beispiel: var Wert = (10 * 20) + (100 - 10); // identisch mit (10 * 20) + 100 - 10;

Schritt 1:	10 * 20	200
Schritt 2:	100 - 10	90
Schritt 3:	200 + 90	290

3.5.12. Operatoren in Javascript 1.5 im Netscape 6.x

arithmetische Operatoren:



+
++
-
--
*
/
%

String-Operatoren:

+
+=

Logische Operatoren:

&&
||
!

Bitweise Operatoren:

&
^
|
~
<<
>>
>>>

Zuweisungsoperatoren:

=
+=
-=
*=
/=
%=
&=
^=
|=
<<=
>>=
>>>=

Vergleichsoperatoren:

==
!=
===
!==
>
>=
<
<=

Sonstige Operatoren:

?:
, (Aufzählung)
delete
function
in
instanceof
new
this
typeof
void

3.6. Anweisungen

mit Semikolon abschließen (außer Blockanweisung und Kommentar). Der Parser ist beim Weglassen des Semikolons nur z.T. fehlerneutral.
Hinweis: Bei Syntaxbeschreibungen bedeutet [] eine Optionalkodierung.



Kommentar

wird nicht geparkt

// das ist ein einzeiliger Kommentar

/*

das ist ein -
oder mehrzeiliger
Kommentar

*/

für JScript: siehe auch bedingtes Parsen durch @ Statements wie @if etc.

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

Bsp.: var Kette = 'test'; + '.jpg'; // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen +
 alert(Kette); // zeigt nur test an und **nicht** test.jpg

Bsp: for (var i = 0; i < 3; i++); // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen {
 {alert(i);} // genau eine Anzeige, die 3 anzeigt

3.6.1. Anweisungen in Javascript und JScript

; Leeranweisung, die nichts bewirkt und nur als Platzhalter dient

anwenden um zu verhindern, dass der Parser einen Teil des Codes nicht berücksichtigt, wenn letzterer ansonsten leer ist

Bsp:

```
if (10 > 9)
{ ; }
else
{ alert("kleiner"); }
```

{ }

Blockanweisung (.z.Z. in der Syntaxvorschrift enthalten)

endet ohne Semikolon

enthält mindestens 1 Anweisung (auch Leeranweisung)

Anweisung(en) mit Semikolon abschließen

break

beendet Schleifen-Anweisung bzw. per Label markierte Anweisung

Achtung: Die Verwendung von break in Schleifen

zeugt von der Faulheit des Programmierers, denn jede Schleife kann ohne break-Anweisung programmiert werden

bedarf einer intern-erweiterten Zeigerverwaltung durch den Browser

siehe label Anweisung und Schleifen und switch Anweisung

Syntax:

break [label];

label Label (Sprungmarke) der abzubrechenden Anweisung

Beispiel 1:

```
var i = 0;
while (i < 100)
{
    if (i == 55)
    {
        break;     // mit alert() weitermachen
    }
    i++;
}
alert("Ende der Schleife");
```

Beispiel 2:

```
var Index = "";
var Menge = {"a": "Athen", "b": "Berlin", "c": "Paris", "d": "Kairo"};
var Kette = "";
```

```
SchleifenAnweisung :                   // ein Label (Marke)
{
    for (Index in Menge)
    {
        Kette = "Hauptstadt von ";

        switch (Menge[Index])
        {
```



```

        case "Berlin":    {
                                Kette += "Deutschland: " + Menge[Index];
                                // nicht weiter auf Athen und Kairo prüfen
                                break;
                            }

        case "Athen":     {
                                Kette += "Griechenland: " + Menge[Index];
                                // kein break, also noch auf Kairo prüfen
                            }

        case "Kairo":     {
                                Kette += "Ägypten: " + Menge[Index]; }

        default:          {
                                // im Falle von Paris:
                                //      Kairo wird nie angezeigt, da im Feld
                                //      hinter Paris

                                break SchleifenAnweisung;
                            }
    }

    {alert(Kette);}      // nicht bei Paris abgearbeitet
}

```

continue

Start des nächsten Schleifendurchlaufes
 alle Anweisungen hinter continue werden ignoriert, wenn sie in der Schleife liegen
 siehe label Anweisung

Syntax:

```
continue [label];
```

label Label (Sprungmarke) der Anweisung z.B. Marke in der Switch-Anweisung, aber der fortgesetzt werden soll
 siehe Anweisung label

Beispiel 1:

```

var Kette= "", i=0;

while (i < 10)
{
    i++;

    // wenn 5 dann Kette nicht erweitern um i
    if (i==5)
    {continue; }

    Kette += i; // wird nicht abgearbeitet, wenn i == 5
}

alert (Kette);      // Ziffer 5 fehlt

```

Beispiel 2:

```

var Feld = new Array();

Outer:   for (var i = 0; i < 5; i++)
{
    Inner:   for (var j = 0; j < 5; j++)
    {
        if (j == 2)
        {continue Inner;} // Sprung zur Marke Inner
        else
            { Feld[i,j] = j + 1; }
    }
}

```

do...while

tue etwas (do), solange (while) es erlaubt ist (while liefert true)
 mindestens 1 Durchlauf
 siehe auch while Anweisung, wenn der Mindestdurchlauf nicht erwünscht ist

Syntax:




```

        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

for...in

tue etwas, wenn es erlaubt ist

Syntax:

```

for (variable in [zeiger])
{ statements }

```

variable	mit Datentyp laut object bzw. array Inhalt
	wenn in der Menge der Werten laut object bzw. array enthalten, so statements abarbeiten
	Menge der Werte: Anzahl der Werte in der Menge ist die maximale Anzahl der
	Schleifenabarbeitung
zeiger	auf Object oder Array
statements	wenn nur 1 Statement so kann Blockanweisung { } entfallen

Beispiel 1:

```

Index, WerteKette = "";
var Menge = { "a" : "Athen", "b" : "Belgrad", "c" : "Kairo" };

for (Index in Menge)
{
    WerteKette += Menge[Index];
}

```

Beispiel 2:

```

function ErzeugeObjekt(Wert1, Wert2)
{
    this.ObjektWert1=Wert1;
    this.ObjektWert2=Wert2;
}

var MeinObjekt = new ErzeugeObjekt(1,2);
for (i in MeinObjekt)
{ alert('Element ' + i + ' ist ' + MeinObjekt[i] + '\n'); }

```

function

Deklaration einer frei-programmierten (privaten) Funktion in Script
Verschachtelung möglich
siehe auch Script-Objekt Function

Syntax:

```

function freier_funktions_bezeichner ([ argumenten_liste ])    // Kopf der Funktion
{ statements }                                                  // Rumpf der Funktion

```

freier_funktions_bezeichner	darf kein Schlüsselwort von Script sein
statements	Rumpf der Funktion immer Blockanweisung kodieren muss nicht return-Anweisung enthalten, kann aber



Hinweis: Eine PROCEDURE gibt es nicht in Script

argumenten_liste

siehe auch Script-Objekt Function

Folge von per Komma getrennten Elementen

Listenelement:

immer Referenz

Platzhalter für Parameter (Wert oder Referenz)

Hinweis: intern wird Wert-Parameter auch referenziert

Wert: z.B. numerisch, Boolean, String, Ausdruck
wird in statements verarbeitet
ist nur funktions-lokal gültig

```
var funktions_name = new function(["argumenten_liste"],"javascript_anweisungen");
```

javascript_anweisungen

sind der Funktionskopf
in " " bzw. ' ' zu setzen
mit ; trennen

Beispiel 1 für Anweisung function:

```
var GlobaleVariable1 = "Bitte laut";
var GlobaleVariable2 = "lein";
var GlobaleVariable3 = "";
```

```
function GlobaleFunktion_ZeichenLiefern(ZeichenArt)           // ZeichenArt ist funktionslokal
{
    var FunktionsLokaleVariable = ""                        // Initialisierung des String

    if (ZeichenArt = "Blank")
    { FunktionsLokaleVariable = " ";}

    if (ZeichenArt = "Doppelpunkt")
    { FunktionsLokaleVariable = ":";}

    return FunktionsLokaleVariable; // Funktion liefert Inhalt von FunktionsLokaleVariable
                                   // und keinen Zeiger auf FunktionsLokaleVariable,
                                   // da ein Zeiger auf eine funktionslokale Variable
                                   // niemals lieferbar ist: Zeiger existiert nur zur Laufzeit
                                   // der Funktion !!
}
```

```
function Globale_BeispielFunktion(           FunktionsLokalesArgument1_Referenz,
                                           FunktionsLokalesArgument2_Wert_Numerisch
                                           FunktionsLokalesArgument2_Wert_String
                                           )
```

```
{
    function LokaleFunktion_LeerZeichenLiefern()
    { return (GlobaleFunktion_ZeichenLiefern("Blank")); }           // liefert " "

    function LokaleFunktion_DoppelpunktLiefern()
    { return GlobaleFunktion_ZeichenLiefern("Doppelpunkt ");}       // liefert ":"

    var FunktionsLokaleVariable1 = "kleine";

    var FunktionsLokaleVariable2 =

        // Referenz auf GlobaleVariable1
        FunktionsLokalesArgument1_Referenz           // "Bitte laut"

        + LokaleFunktion_LeerZeichenLiefern()           // "Bitte laut "

        + "mitsingen"                                   // "Bitte laut mitsingen"

        + LokaleFunktion_LeerZeichenLiefern()           // "Bitte laut mitsingen "

        + LokaleFunktion_DoppelpunktLiefern()           // "Bitte laut mitsingen : "

        + LokaleFunktion_LeerZeichenLiefern()           // "Bitte laut mitsingen : "
```



```

// numerischen Wert 3 konvertieren zu "3"
+ FunktionsLokalesArgument2_numerisch_Wert.toString()
// "Bitte laut mitsingen : 3"

+ LokaleFunktion_LeerZeichenLiefern()
// "Bitte laut mitsingen : 3 "

// "kleine"
+ FunktionsLokaleVariable
// "Bitte laut mitsingen : 3 kleine"

+ LokaleFunktion_LeerZeichenLiefern()
// "Bitte laut mitsingen : 3 kleine "

// String-Wert "Neger"
+ FunktionsLokalesArgument2_Wert_String;
// "Bitte laut mitsingen : 3 kleine Neger"

GlobaleVariable3 = FunktionsLokaleVariable2;
// Inhalt der lokalen Variablen nach
// globale Variable kopieren
// man hätte auch return-Anweisung
// nehmen können
}

// Aufruf mit korrekter Argumentenversorgung durch Parameterliste also
// GlobaleVariable1 "Bitte laut"
// Ausdruck 1 + 2 3
// "Neger"
// Funktion belegt GlobaleVariable3 mit dem Wert "Bitte laut mitsingen : 3 kleine Neger"

Globale_BeispielFunktion(
    GlobaleVariable1, // "Bitte laut"
    1 + 2, // Ausdruck, der den Wert 3 liefert
    "Neger"
);

// Anzeige per Alert-Box (Standard-Funktion) von "Bitte laut mitsingen : 3 kleine Negerlein ..."
alert(
    GlobaleVariable3
// "Bitte laut mitsingen : 3 kleine Neger"

+ GlobaleVariable2
// "Bitte laut mitsingen : 3 kleine Negerlein"

+ GlobaleFunktion_ZeichenLiefern("Blank") // "Bitte laut mitsingen : 3 kleine Negerlein "
// Achtung: Aufruf von LokaleFunktion_DoppelpunktLiefern()
// ist nicht zulässig

+ "..."
// "Bitte laut mitsingen : 3 kleine Negerlein ..."
);

```

Beispiel 2 für Ableitung vom Script-Objekt Function :

```
var Zeiger = new Function("return init_wert;")
```

Argumente einer Funktion:

```

Beispiel: function test(x)
{alert(x==null);}

test(); // liefert true
test(1); // liefert false

```

Werden Argumente nicht mit Wert belegt, so werden die Variablen der Argumente nicht erzeugt, sind also null.

Argumente sind null-Zeiger, wenn kein Wert mit Funktionsaufruf übergeben wird.

Es müssen also bei korrekter Programmierung die Argumente auf geprüft werden
ERST auf != null
DANN auf Wertbereich.

Achtung: Der Datentyp des Argumentes wird mit dem Wert, dem das Argument erhält, festgelegt.

Das Argument ist ein Zeiger, der auf einen Speicherbereich mit einem Wert, der Daten nach JScript-zulässigen Typen hat.

:



Returnwert einer Funktion:

Returnwert: ist optionale Referenz auf Rückgabewert

Auch wenn Returnwert geliefert wird, so muss er nicht verarbeitet werden.

Beispiel 1:

```
function test()
{ var X=3;}

alert(test()!=null);    // liefert false, keine Referenz
alert(test());          // liefert undefined da keine Referenz
```

Beispiel 2:

```
function test()
{ var X=3;return X;}

alert(test()!=null);    // liefert true, Referenz vorhanden und ausgewertet
alert(test());          // liefert 3 da alert die Referenz auswertet
```

Beispiel 3:

```
function test()
{ var X=3;return X;}

test();                // kein Scriptfehler, aber Referenz nicht ausgewertet
```

if...else

Fallabfrage

siehe auch Operator ?:

Verschachtelung möglich

Syntax:

```
if (condition)
{ statements1 }
[else { statements2 }]
```

condition

Ausdruck der true oder false liefert
Referenz auf Boolean-Variable
wenn true, so statements1 abgearbeitet
wenn false, so statements2 abgearbeitet

statements1 bzw. 2 wenn nur 1 Statement **nur dann** darf Blockanweisung { } entfallen

Beispiel:

```
var x=5;
var y=8;
var z=0;

// aktuelles x prüfen
if (x == 5)
{
    // x ist 5
    // aktuelles y prüfen
    if (y == 6)
    {
        // y ist 6
        z = 17;
    }
    else
    {
        // y ist nicht 6
        z = 20;
        x = 6;
    }
}
else
{ x = 0; } // Achtung: x ist inzwischen auf 6, wird aber für else nicht verwendet
// aber x wird auf 0 gesetzt !!
```

in

prüfen ob String in einem Objekt als Menge von Strings enthalten ist

in Operator arbeitet analog in der Anweisung for in

Syntax:

[var Wert =] ausdruck_oder_referenz **in** referenz_auf_objekt_mit_string_elementen



ausdruck	muss String liefern
referenz	auf String-Variable
Wert	true, so enthalten false, so nicht enthalten

Beispiel für Array Objekt aus Literalen:

```
var Menge = { "a" : "Athen", "b" : "Belgrad", "c" : "Kairo" };

var Index = "a";
if( "a" in Menge )
{ alert(Menge[Index]); }
```

Label

Sprungmarke für die continue Anweisung innerhalb Schleife
 Marke für die break Anweisung zum Abbrechen der mit Label markierten Anweisung
 siehe auch Anweisungen break und continue
 Syntax:

```
label : { statements }
```

label	anstelle von "label" ist ein freier Bezeichner zu kodieren, der kein Bezeichner eines bereits definierten Elementes und kein Schlüsselwort sein darf
-------	--

statements	wenn nur 1 Statement so kann Blockanweisung { } entfallen
------------	---

Beispiel 1:

```
var Feld = new Array();

Outer:   for (var i = 0; i < 5; i++)
{
    Inner:   for (var j = 0; j < 5; j++)
    {
        if (j == 2)
        { continue Inner; } // Sprung zur Marke Inner
        else
        { Feld[i,j] = j + 1; }
    }
}
```

Beispiel 2:

```
var Index = "";
var Menge = { "a" : "Athen", "b" : "Berlin", "c" : "Paris", "d" : "Kairo" };
var Kette = "";

SchleifenAnweisung :           // ein Label (Marke)
{
    for (Index in Menge)
    {
        Kette = "Hauptstadt von ";

        switch (Menge[Index])
        {
            case "Berlin":      {
                                    Kette += "Deutschland: " + Menge[Index];
                                    // nicht weiter auf Athen und Kairo prüfen
                                    break;
                                }

            case "Athen":       {
                                    Kette += "Griechenland: " + Menge[Index];
                                    // kein break, also noch auf Kairo prüfen
                                }

            case "Kairo":       {
                                    Kette += "Ägypten: " + Menge[Index]; }

            default:            {
                                    // im Falle von Paris:
                                    // Kairo wird nie angezeigt, da im Feld
                                    // hinter Paris

                                    break SchleifenAnweisung;
                                }
        }
    }
}
```



```

    }
    {alert(Kette);} // nicht bei Paris abgearbeitet
  }
}

```

new

Objekt (Objektinstanz) erzeugen
 Zeiger auf Instanz erzeugen und Speicher reservieren (allokieren)
 Prototyping einer Instanz ist möglich
 es sind auch Instanzen von Script-Objekten erzeugbar (Konstruktor ist der Bezeichner des Script-Objektes)
 z.B.

Objekt arguments
 Objekt Array
 Objekt Boolean
 Objekt Date
 Objekt Enumerator
 Objekt Error
 Objekt Function
 Objekt Math
 Objekt Number
 Objekt Object (nicht Objekt object des Internet Explorer für das HTML-Tag OBJECT)
 Objekt RegExp
 Objekt String
 Objekt var

Achtung: Der Browser kann nur Objekte verarbeiten, die er kennt, z.B. ein Array Objekt, dessen Methoden und Eigenschaften dem Browser bekannt sind. Bei einem privaten Objekt müssen alle Eigenschaften und Methoden auf Script-Komponenten bestehen.

Jedes Objekt kann per Prototyping um Eigenschaften und Methoden erweitert werden, wenn es die Eigenschaft `.prototype` besitzt.

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft `.prototype` erzeugt !

Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft `.prototype`.

Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft `.prototype`.

Punktnotation zum Zeiger:

```

zeiger.prototype.eigenschaft
zeiger.prototype.methode

```

Erweiterung eines Script-Objektes per

```

bezeichner_script_objekt.prototype.eigenschaft
bezeichner_script_objekt.prototype.methode

```

Eigenschaften und Methoden müssen auf Script-Elemente **basieren**, denn nur letztere kennt die Scriptmaschine des Browsers.

Syntax:

```
[ var Zeiger = ] new bezeichner [( [ArgumentenListe] )]
```

ArgumentenListe siehe Beschreibungen der einzelnen Objekte
 Hinweis: Script-Objekte werden in der alphabetisch-sortierten Beschreibung der Objekte und Collectionen (z.T. browser-spezifisch) beschrieben.

Zeiger wenn **null** (nicht numerisch 0 !!), so konnte das Objekt nicht erzeugt werden

Beispiele:

```

var InstanzVomTyp_Object = new Object;

var InstanzVomTypArray = new Array();

var InstanzVomTypeDate = new Date("Jan 5 1996");

```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:

Es wird die Eigenschaft `.prototype` **nicht** erzeugt !

```

<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)

```



```

    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
    this.nummer = nummer;
    this.plz = plz;
    this.ort = ort;
    this.methode = methode;
}

person = new datenstruktur_erzeugen
(
    "Erika",           // Vorname
    "Mustermann",      // Nachname
    "Musterstrasse",   // Strasse
    "1",               // Nummer
    "10000",           // PLZ
    "Musterstadt",     // Ort
    datenstruktur_ausgeben // ohne () kodieren
);

// alternativ auch kodierbar:
// var person = {
//     vorname:"Erika",           // Vorname
//     nachname:"Mustermann",     // Nachname
//     strasse:"Musterstrasse",   // Strasse
//     nummer:"1",               // Nummer
//     plz:"10000",              // PLZ
//     ort:"Musterstadt",        // Ort
//     methode:datenstruktur_ausgeben// Ausgabemethode ohne () kodieren
// };

// Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !
// -->
</SCRIPT>
</HTML>

```

Beispiel für Ableitung aus einem JScript-Objekt:

```

var Kette = new String("Hi");
alert( (Kette.constructor == String)); // liefert "true"
alert( (Kette.constructor == "String")); // liefert "false"

```

Beispiel für Ableitung anhand privaten Konstruktors:

```

function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion(); // bewirkt Ausführung von TestFunktion() also auch von alert()

// ZeigerAufFunktion(); // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
//                       // da keine Ableitung vom JScript-Objekt Function
// Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration erkannt

```



```

    alert(ZeigerAufFunktion.constructor == TestFunktion); // true
    alert(TestFunktion.constructor == TestFunktion);      // false

```

return

Funktionsergebnis liefern
 letzte Zeile innerhalb einer Funktion
 Hinweis: Funktion muss kein return besitzen
 siehe Anweisung function und Objekt Function

Syntax:

```
return ([ausdruck]);
```

```
return [ausdruck];
```

ausdruck enthält zu lieferndes Funktionsergebnis
 wenn nicht kodiert, wo wird undefined bzw. null geliefert

Beispiel:

```

var GlobaleVariable1 = "Bitte laut";
var GlobaleVariable2 = "lein";
var GlobaleVariable3 = "";

```

```

function GlobaleFunktion_ZeichenLiefern(ZeichenArt)    // ZeichenArt ist funktionslokal
{
    var FunktionsLokaleVariable = ""                  // Initialisierung des String

    if (ZeichenArt = "Blank")
    { FunktionsLokaleVariable = " "; }

    if (ZeichenArt = "Doppelpunkt")
    { FunktionsLokaleVariable = ":"; }

    return FunktionsLokaleVariable; // Funktion liefert Inhalt von FunktionsLokaleVariable
                                   // und keinen Zeiger auf FunktionsLokaleVariable,
                                   // da ein Zeiger auf eine funktionslokale Variable
                                   // niemals lieferbar ist: Zeiger existiert nur zur Laufzeit
                                   // der Funktion !!
}

```

```

function Globale_BeiispielFunktion(    FunktionsLokalesArgument1_Referenz,
                                       FunktionsLokalesArgument2_Wert_Numerisch
                                       FunktionsLokalesArgument2_Wert_String
                                       )
{

```

```

    function LokaleFunktion_LeerZeichenLiefern()
    { return (GlobaleFunktion_ZeichenLiefern("Blank")); }    // liefert " "

```

```

    function LokaleFunktion_DoppelpunktLiefern()
    { return GlobaleFunktion_ZeichenLiefern("Doppelpunkt ");}    // liefert ":"

```

```
var FunktionsLokaleVariable1 = "kleine";
```

```
var FunktionsLokaleVariable2 =
```

```

    // Referenz auf GlobaleVariable1
    FunktionsLokalesArgument1_Referenz    // "Bitte laut"

```

```
+ LokaleFunktion_LeerZeichenLiefern()    // "Bitte laut "
```

```
+ "mitsingen"    // "Bitte laut mitsingen"
```

```
+ LokaleFunktion_LeerZeichenLiefern()    // "Bitte laut mitsingen "
```

```
+ LokaleFunktion_DoppelpunktLiefern()    // "Bitte laut mitsingen :"
```

```
+ LokaleFunktion_LeerZeichenLiefern()    // "Bitte laut mitsingen : "
```

```

    // numerischen Wert 3 konvertieren zu "3"
    + FunktionsLokalesArgument2_numerisch_Wert.toString()    // "Bitte laut mitsingen : 3"

```

```
+ LokaleFunktion_LeerZeichenLiefern()    // "Bitte laut mitsingen : 3 "
```



```

// "kleine"
+ FunktionsLokaleVariable // "Bitte laut mitsingen : 3 kleine"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen : 3 kleine "

// String-Wert "Neger"
+ FunktionsLokalesArgument2_Wert_String; // "Bitte laut mitsingen : 3 kleine Neger"

GlobaleVariable3 = FunktionsLokaleVariable2; // Inhalt der lokalen Variablen nach
// globale Variable kopieren
// man hätte auch return-Anweisung
// nehmen können
}

// Aufruf mit korrekter Argumentenversorgung durch Parameterliste also
// GlobaleVariable1 "Bitte laut"
// Ausdruck 1 + 2 3
// "Neger"
// Funktion belegt GlobaleVariable3 mit dem Wert "Bitte laut mitsingen : 3 kleine Neger"

Globale_BeispielFunktion(
    GlobaleVariable1, // "Bitte laut"
    1 + 2, // Ausdruck, der den Wert 3 liefert
    "Neger"
);

// Anzeige per Alert-Box (Standard-Funktion) von "Bitte laut mitsingen : 3 kleine Negerlein ..."
alert(
    GlobaleVariable3 // "Bitte laut mitsingen : 3 kleine Neger"

+ GlobaleVariable2 // "Bitte laut mitsingen : 3 kleine Negerlein"

+ GlobaleFunktion_ZeichenLiefern("Blank") // "Bitte laut mitsingen : 3 kleine Negerlein "
// Achtung: Aufruf von LokaleFunktion_DoppelpunktLiefern()
// ist nicht zulässig

+ "..." // "Bitte laut mitsingen : 3 kleine Negerlein ..."
);

```

switch

Auswahl
siehe Anweisung break

Syntax:

```

switch (referenz)
{
    case erster_wert : { statements }

    .....

    case letzter_wert : { statements }

    default : { statements }
}

```

referenz auf eine Variable etc.

erster_wert ... letzter_wert zu referenz typengerechte **Werte**, anhand denen ausgewählt wird
beliebig viele Wert möglich

Wertauswahl in der Kodierungsfolge innerhalb switch

und wenn referenz den Wert hat

Achtung: Es werden **alle** entsprechenden Werte ausgewählt

aber: siehe statements mit break-Anweisung

default: nur dann abgearbeitet, wenn kein einzigstes Label auswählbar war

aber: siehe statements mit break-Anweisung

statements

optional (Empfehlung: mindesten Leeraanweisung kodieren)

wenn nur 1 Statement **dann** darf Blockanweisung { } entfallen

muss Anweisung break enthalten als letzte Anweisung, wenn

ausgeschlossen werden soll, dass keine weiteren nachfolgenden

Vergleiche der referenz mit Werten erfolgen **und auch nicht**

default abgearbeitet werden sollen.



Beispiel:

```
var Index = "";
var Menge = {"a" : "Athen", "b" : "Berlin", "c" : "Paris", "d" : "Kairo"};
var Kette = "";

SchleifenAnweisung :           // ein Label (Marke)
{
    for (Index in Menge)
    {
        Kette = "Hauptstadt von ";

        switch (Menge[Index])
        {
            case "Berlin":      {
                                   Kette += "Deutschland: " + Menge[Index];
                                   // nicht weiter auf Athen und Kairo prüfen
                                   break;
                               }

            case "Athen":       {
                                   Kette += "Griechenland: " + Menge[Index];
                                   // kein break, also noch auf Kairo prüfen
                               }

            case "Kairo":       {
                                   Kette += "Ägypten: " + Menge[Index]; }

            default:           {
                                   // im Falle von Paris:
                                   //      Kairo wird nie angezeigt, da im Feld
                                   //      hinter Paris
                                   break SchleifenAnweisung;
                               }
        }

        { alert(Kette); }      // nicht bei Paris abgearbeitet
    }
}
```

this

Zeiger auf das aktuelle Objekt für Punktnotation
eigentlich keine Anweisung, da nicht mit Semikolon zu beenden ist

try catch finally

Fehlerbehandlung in Script (nicht Ereignisse von Objekten !)
Verschachtelung möglich
siehe auch error JScript-Objekt des Internet Explorer für private Run-Time-Error

Fehler: Runtime Error
oder per throw Anweisung erzeugter Error

Syntax:

```
try          {tryStatements}
[catch(exception) {catchStatements}]
[finally     {finallyStatements}]
```

tryStatements	können Fehler erzeugen, der abgefangen werden soll
exception	freier Bezeichner als Platzhalter für den Fehler, der aus der Abarbeitung von tryStatements resultiert Variable wird automatisch gefüllt
catchStatements	exception auslesen und daraufhin eine Reaktion ausführen werden nur abgearbeitet, wenn Fehler auch wirklich auftrat in der Abarbeitung von tryStatements
finallyStatements	wird immer abgearbeitet, egal ob ein Fehler in der Abarbeitung von tryStatements und / oder catchStatements auftrat oder nicht

Beispiel:

```
function Anzeige(Kette)
```



```

    { alert(Kette); }

    try { Anzeige("try1");
    {
        try { Anzeige("try 2"); }
        catch(e) { Anzeige("catch2 " + e); }
        finally { Anzeige("finally2"); }
    }
    catch(e) { Anzeige("catch1 " + e); }
    finally { Anzeige("finally1"); }

```

Beispiel:

```

X04=false;
// +++++ CLASSID per try-catch zuweisen
try{ X02.classid=X00;}catch(e){X04=true;}           // irgendwas tun
                                                    // e ist Platzhalter für Fehlercode, der aber nicht ausgewertet

```

wird

```

// +++++ classid-Belegung prüfen
if(X04){.....}                                   // anstelle e wird X04 ausgewertet

```

throw

freie Fehlerbedingung erzeugen für try...catch...finally

throw im try kodieren als freie Fehlerbedingung
catch auswerten

Syntax:

```
throw exception;
```

exception Wert oder Variable

Beispiel 1:

throw "Error2";	generiert userdefinierte Ausnahme "Error2"	als String
throw 42;	generiert userdefinierte Ausnahme 42	als numerischer Wert
throw true;	generiert userdefinierte Ausnahme true	als Boolean

Beispiel 2:

```

function Anzeige(Kette)
{ alert(Kette); }

try { Anzeige("try1");
{
    try
    {
        throw "Das ist eine Fehlerbedingung";
        Anzeige("try 2");
    }
    catch(e)
    {
        if ( e == "Das ist eine Fehlerbedingung" )
        {Anzeige("catch2 " + e); }
    }
    finally { Anzeige("finally2"); }
}
catch(e) { Anzeige("catch1 " + e); }
finally { Anzeige("finally1"); }

```

Beispiel 3:

```

function ErzeugeAusnahme Bedingung(Ausnahme Bedingung)
{
    this.Ausnahme Bedingung=Ausnahme Bedingung;
    this.AusnahmeArt="UserException";
}

function HoleMonatAlsString (MonatsNummer) // liefert Monat als String
{
    var Index = MonatsNummer -1;

    var MonatsFeld =new Array("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

    if (MonatsFeld[Index] != null)
    {return MonatsFeld[Index];}
    else
    {
        var UserDefinierteAusnahmeBedingung =
        new ErzeugeAusnahme Bedingung ("FalscheMonatsNummer");
    }
}

```




```

        throw UserDefinierteAusnahmeBedingung;
    }
}

```

var
Deklaration einer Variablen
siehe auch Objekt var
Syntax:

```
var variable1 [ = value1 ] [, variablenliste [ = value2]];
```

variablenliste Bezeichner mit Komma trennen
alle Variablen erhalten einen gemeinsamen Wert laut value2

Beispiele:

```
var index;
var name = "Test";
var answer = 42, counter, numpages = 10;
```

while
true etwas (do), solange (while) es erlaubt ist (also solange while den Wert true liefert)
siehe auch do .. while Anweisung, wenn mindestens 1 Durchlauf erwünscht ist
Syntax:

```
while (ausdruck) ;
{ statements }
```

statements wenn nur 1 Statement so kann Blockanweisung { } entfallen

ausdruck muss true oder false liefern
wenn true, so nächster Schleifendurchlauf
wenn false, so Ende der Schleife

with

Instanz auf eine Anweisung zuordnen, die die Instanz verarbeiten soll

Syntax:

with (referenz)

{ statements }

statements

wenn nur 1 Statement so kann Blockanweisung { } entfallen

Beispiel:

```

with (Math)
{
    // alle Methoden entstammen dem Script-Objekt Math
    var x = cos(3 * PI) + sin (LN10) ;
    var y = tan(14 * E);
}

```

3.6.2. Anweisungen nur in Microsoft JScript**@cc_on** und **@end**

Container für bedingtes Parsen per Scriptmaschine

immer innerhalb

/* @*/

oder nach //

also innerhalb eines Kommentars kodieren, falls Browser nicht IE ist

bei // zwischen // und @ darf **kein Leerzeichen** liegen !

Empfehlung: kein Blank nach /* bzw. vor @*/

siehe auch @if und @set

Beispiel:

```

<SCRIPT>
/* @cc_on */
/* @if ( @_jscript_version >= 4 )
    {alert("JScript ab Version 4 oder höher");}
    @else @*/

    // nachfolgende Meldung kommt für alle Browser, die nicht bedingt parsen können
    alert("kein IE oder JScript unter Version 4, also kein bedingtes Parsen möglich");

/* @end */
</SCRIPT>

```

vordefinierte Variablen für das bedingte Parsen:

werden automatisch belegt mit true oder NaN



@_win32	true, so	Win32-System ist aktiv
@_win16	true, so	Win16-System ist aktiv
@_mac	true, so	Apple Macintosh-System aktiv
@_alpha	true, so	DEC Alpha Prozessor aktiv
@_x86	true, so	Intel Prozessor aktiv
@_mc680x0	true, so	Motorola 680x0 Prozessor aktiv
@_PowerPC	true, so	Motorola PowerPC Prozessor aktiv
@_jscript		immer true, da bedinge Kompilierung nur mit JScript läuft
@_jscript_build		Buildnummer der JScript-Scriptmaschine
@_jscript_version		Version der JScript-Scriptmaschine
	Aufbau	major_nummer.minor_nummer

@if und @elif und @else und @end

Analogon zum Operator ?: und zur if-Anweisung
immer innerhalb

```
/* @*/
oder nach //
also innerhalb eines Kommentars kodieren, falls Browser nicht IE ist

bei // zwischen // und @ darf kein Leerzeichen liegen !
```

Empfehlung: kein Blank nach /* bzw. vor @*/

Syntax:

```
@if (condition1)
script1
    [@elif (condition2) script2]
[@else script3]
@end
```

condition1 und 2 müssen boolean liefern

script1 bis 3 werden bedingt ausgeführt

@elif auch mehrfach kodierbar, aber alle **vor** einem eventuellen @else

Beispiel:

```
<SCRIPT>
/* @cc_on */
/* @if ( @_jscript_version >= 4 )
    {alert("JScript ab Version 4 oder höher");}
    @else
/* nachfolgende Meldung kommt für alle Browser, die nicht bedingt parsen können
    alert("kein IE oder JScript unter Version 4, also kein bedingtes Parsen möglich");

/* @end */
</SCRIPT>
```

vordefinierte Variablen für das bedingte Parsen:
werden automatisch belegt mit true oder NaN

@_win32	true, so	Win32-System ist aktiv
@_win16	true, so	Win16-System ist aktiv
@_mac	true, so	Apple Macintosh-System aktiv
@_alpha	true, so	DEC Alpha Prozessor aktiv
@_x86	true, so	Intel Prozessor aktiv
@_mc680x0	true, so	Motorola 680x0 Prozessor aktiv
@_PowerPC	true, so	Motorola PowerPC Prozessor aktiv
@_jscript		immer true, da bedinge Kompilierung nur mit JScript läuft
@_jscript_build		Buildnummer der JScript-Scriptmaschine
@_jscript_version		Version der JScript-Scriptmaschine
	Aufbau	major_nummer.minor_nummer

@set

Erzeugung und Belegung einer globalen Variable (numerischer oder Boolean-Wert)



Verwendung und Operationen mit der Variablen **nur** innerhalb des bedingten Parsens möglich
immer innerhalb

/* @*/
 oder nach //
 also innerhalb eines Kommentars kodieren, falls Browser nicht IE ist

bei // zwischen // und @ darf **kein Leerzeichen** liegen !

Empfehlung: kein Blank nach /* bzw. vor @*/

mögliche Operatoren sind:

! ~ * / % + - << >> >>> < <= > >= == != === !== & ^ | && ||

Bezug auf eine nicht erzeugte Variable liefert immer NaN und keinen Fehler

Bsp: @if (@NichtexistenteVariable == NaN)

@if (@newVar != @newVar) ist zulässig und prüft auf NaN

Syntax:

@set @varname = term

varname Aufbau des Namens wie bei Javascript-Bezeichnern

term Ergebnis bzw. Inhalt des Terms wird der Variablen zugewiesen
 darf nicht String liefern
 kann nur numerisch oder boolean sein

= Zuweisungsoperator

Beispiele:

@set @myvar1 = 12;

@set @myvar2 = (@myvar1 * 20);

@set @myvar3 = @_jscript_version;

vordefinierte Variablen für das bedingte Parsen:

werden automatisch belegt mit true oder NaN

@_win32	true, so	Win32-System ist aktiv
@_win16	true, so	Win16-System ist aktiv
@_mac	true, so	Apple Macintosh-System aktiv
@_alpha	true, so	DEC Alpha Prozessor aktiv
@_x86	true, so	Intel Prozessor aktiv
@_mc680x0	true, so	Motorola 680x0 Prozessor aktiv
@_PowerPC	true, so	Motorola PowerPC Prozessor aktiv
@_jscript		immer true, da bedingte Kompilierung nur mit JScript läuft
@_jscript_build		Buildnummer der JScript-Scriptmaschine
@_jscript_version		Version der JScript-Scriptmaschine
		Aufbau major_nummer.minor_nummer

3.6.3. Anweisungen in Javascript 1.5 im Netscape 6.x

break

const Deklaration einer Variable mit konstantem Inhalt, der nur lesbar ist
 Bsp: const a = 7;

continue

do...while

export Methoden, Eigenschaften, Objekte und Funktionen eines signierten Scripts für ein anderes unsigniertes oder signiertes Script verfügbar machen, das import-Anweisung besitzen muss.

Syntax:

export liste
 oder export *

liste kommagetrennte Bezeichner von Methoden, Eigenschaften, Objekten und Funktionen

* für alle Methoden, Eigenschaften, Objekte und Funktionen im signierten Script

Beispiel:

zu exportieren sind im signierten Script die Eigenschaften f und p von obj



signierte Script: export obj.f, obj.p

anderes Script: import obj.f, obj.p

Hinweis: export obj schließt alle Eigenschaften etc. ein.

for
for...in
function
if...else
import

Methoden, Eigenschaften, Objekte und Funktionen eines signierten Script, die dort per export für ein anderes unsigniertes oder signiertes Script verfügbar gemacht wurden, importieren

Syntax:

import liste
oder import *

liste kommagetrennte Bezeichner von Methoden, Eigenschaften, Objekten und Funktionen

* für alle Methoden, Eigenschaften, Objekte und Funktionen aus dem signierten Script

Beispiel:

zu exportieren sind im signierten Script die Eigenschaften f und p von obj

signierte Script: export obj.f, obj.p

anderes Script: import obj.f, obj.p

Hinweis: export obj schließt alle Eigenschaften etc. ein.

label
return
switch
throw
try...catch
var
while
with

Deklaration einer les- und schreibbaren Variablen mit variablem Inhalt

3.7. Ausdruck (Expression)

Kombination aus Variablen, Operatoren, Literalen und Anweisungen
liefert immer Wert oder Referenz (Zeiger)

Beispiele:

```
4 + 5
x += 1
10 / 2
a & b
x++
var Wert = 3 * (4 / 5) + 6;
var Kette = "Test " + Wert.toString() + " !";
var Feld = new Array("Hallo", Math.PI, 42.456);
var FeldElement = Feld[1];
```

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

Bsp.: var Kette = 'test'; + '.jpg'; // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen +
alert(Kette); // zeigt nur test an und **nicht** test.jpg

Bsp: for (var i = 0; i <3; i++); // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen {
{alert(i);} // genau 1 Anzeige, die 3 anzeigt

3.8. Funktion

basiert auf dem Script-Objekt Function und der Anweisung function

Hinweis: Bei Syntaxbeschreibungen bedeutet [] eine Optionalkodierung.

Funktion und Methode

Funktion ist synonym zu Methode, wenn es sich um eine Funktion eines Objektes handelt.

Beispiel für eine Methode eines Objektes:

```
var fenster=window.open( url_der_html_datei,  
name_der_html_datei,
```



```
"toolbar=0,location=0,.....,height=140" // eine Zeile und keine Blanks !
);
```

Funktion vordefiniert oder frei deklariert

Vordefinierte Funktionen liegen als Methoden von Objekten vor.

Der Programmierer kann Funktionen frei deklarieren:

Eine frei deklarierte Funktion kann per Prototyping als Methode eines Objektes verwendet werden.

Beispiel für eine Funktion, die vom Programmierer definiert wurde:

```
function Test(Wert1, Wert2)
{
    var Summe = Wert1 + Wert2;
    alert (Summe.toString());
}
```

Funktion und Funktionswert

Eine Funktion **kann** einen Funktionswert liefern (siehe return Anweisung), muss aber nicht. Das zu kontrollieren, obliegt leider nur dem Programmierer, der also gewissenhaft kodieren muss.

Beispiel:

```
function Test(Wert1, Wert2)
{return ( Wert1 + Wert 2);}
```

Aufruf einer Funktion

Der Aufruf einer Funktion darf nur dort stattfinden, wo eine Referenz zulässig ist, da der Funktionsbezeichner einen Zeiger repräsentiert. Z.B. ist der Aufruf einer Funktion innerhalb eines Literals unmöglich.

Falls die Funktion etwas liefert, gilt zusätzlich:

Anstelle des Funktionsaufrufes wird die gelieferte Größe gesetzt.

Die Funktion muss datentyp-gerecht liefern (entspricht einem typisierten Zeiger).

Analog gilt das auch für den Aufruf einer Methode.

Beispiel für Aufruf einer Funktion innerhalb eines Ausdrucks:

```
function Test(Wert1, Wert2)
{return ( Wert1 + Wert 2);}

var Wert = 20 + Test(10,20);

alert(Wert.toString());

// Nachfolgender Aufruf ergibt einen Fehler, da die Funktion einen numerischen Wert liefert
// Die Methode alert() kann eventuell automatisch nach String konvertieren
alert( "Das Ergebnis lautet " + Test(10,20));
```

Der Aufruf einer Funktion innerhalb eines HTML-Attributes anstelle des Wertes des Attributes ist zulässig, wenn die Funktion etwas liefert.

Beispiel für Aufruf einer Funktion im HREF-Attribut von z.B. <A> und <AREA>:

```
<A HREF="javascript:name_der_funktion(....);">
.....
</A>
```

3.8.1. Funktion und optionale Argumente und Parameter

Eine Funktion kann Argument(e) optional im Funktionskopf kodiert bekommen.

Beispiel:

```
function Test(argument1, argument2) // Funktionskopf
{
    // das ist der Funktionsrumpf
}
```

Vordefinierte Funktionen als Methoden eines Objektes haben in der Regel Argumente.

Argument und Argumentenliste



Argument ist der Platzhalter für den Wert, der per Funktionsaufruf mit dem Parameter übergeben wird
 wird nicht var-deklariert im Funktionskopf (siehe Script-Objekt arguments)
 wird nicht in " " bzw. ' ' kodiert
 darf kein Schlüsselwort sein
 darf keine Funktion sein
 darf kein Ausdruck sein

Argument(e) sind optional kodierbar in der Funktionsdeklaration:
 im Funktionskopf, also innerhalb von ()
 als Liste von kommagetrennten Argumenten

Parameter und Parameterliste

Parameter ist bei Funktionsaufruf der konkrete Wert für den Platzhalter in Form des Argumentes
 wird nicht var-deklariert innerhalb von () beim Funktionsaufruf
 kann Ausdruck sein, der Zeiger oder Wert liefert
 kann Aufruf einer Funktion sein, die per return Anweisung etwas liefern **muss**.
 kann Wert sein z.B. String-Wert etc. (siehe Datentypen und Script-Objekte)
 Wert muss zum Datentyp des Argumentes passen

Parameterliste Folgen von kommagetrennten Werten, die die Argumente füllen:
 Die Folge der Listenelemente entspricht der Argumentenfolge in der Argumentenliste.
 Es muss **jedes** Argument gefüllt werden (keine Lücken).

Aufruf einer Funktion mit Parameterliste

nur möglich, wenn Funktion mit Argument(en) im Funktionskopf deklariert wurde

Parameter werden **automatisch** mit dem Script-Objekt arguments (siehe dort) verarbeitet:

arguments Objekt ist ein Feld
 dient als Schnittstelle für den Programmierer
 Parameter in der Listenfolge von links nach rechts in das Feld automatisch abgelegt
 mit Index ab 0 und aufsteigend
 Anzahl der Parameter laut arguments.length

Bsp: `function test(arg1, arg2, arg3) {.....}`
`.....`
`test(10, 20, 30);`
 10 entspricht `test.arguments[0]` für arg1
 20 entspricht `test.arguments[1]` für arg2
 30 entspricht `test.arguments[2]` für arg3

Anzahl der Argument entspricht `test.arguments.length` und ist 3

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

```
var GlobalesFeld = new Array();

function FunktionsBezeichner()
{
    // Argumenteliste der Funktion lokal zur Funktion referenzieren
    var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

    // Anzahl der Argumente
    var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

    if ( ArgumenteAnzahl > 0 )
    {
        // Arrgumentenliste auslesen
        for (var i = 0; i < ArgumenteAnzahl; i++)
        { GlobalesFeld[i] = ArgumentenListeAlsFeld[i]; }
    }

    // hier die weiteren Anweisungen der Funktion
}
```

3.8.2. Funktion und optionaler Funktionswert

Ein Funktionswert ist optional im Funktionsrumpf kodierbar. Dazu wird die return Anweisung verwendet:



return Anweisung:

Funktionsergebnis liefern
 letzte Zeile innerhalb einer Funktion
 Hinweis: Funktion muss kein return besitzen
 siehe Anweisung function und Script-Funktion

Syntax:

```
return ([ausdruck]);
```

```
return [ausdruck];
```

ausdruck

enthält zu lieferndes Funktionsergebnis
 wenn nicht kodiert, wo wird undefined bzw. null geliefert

Beispiel:

```
var GlobaleVariable1 = "Bitte laut";
var GlobaleVariable2 = "lein";
var GlobaleVariable3 = "";

function GlobaleFunktion_ZeichenLiefern(ZeichenArt)      // ZeichenArt ist funktionslokal
{
    var FunktionsLokaleVariable = ""                  // Initialisierung des String

    if (ZeichenArt = "Blank")
    { FunktionsLokaleVariable = " "; }

    if (ZeichenArt = "Doppelpunkt")
    { FunktionsLokaleVariable = ":"; }

    return FunktionsLokaleVariable; // Funktion liefert Inhalt von FunktionsLokaleVariable
                                   // und keinen Zeiger auf FunktionsLokaleVariable,
                                   // da ein Zeiger auf eine funktionslokale Variable
                                   // niemals lieferbar ist: Zeiger existiert nur zur Laufzeit
                                   // der Funktion !!
}

function Globale_BeiispielFunktion(      FunktionsLokalesArgument1_Referenz,
                                       FunktionsLokalesArgument2_Wert_Numerisch
                                       FunktionsLokalesArgument2_Wert_String
                                       )
{
    function LokaleFunktion_LeerZeichenLiefern()
    { return (GlobaleFunktion_ZeichenLiefern("Blank")); }      // liefert " "

    function LokaleFunktion_DoppelpunktLiefern()
    { return GlobaleFunktion_ZeichenLiefern("Doppelpunkt ");}  // liefert ":"

    var FunktionsLokaleVariable1 = "kleine";

    var FunktionsLokaleVariable2 =

        // Referenz auf GlobaleVariable1
        FunktionsLokalesArgument1_Referenz      // "Bitte laut"

        + LokaleFunktion_LeerZeichenLiefern()      // "Bitte laut "

        + "mitsingen"                              // "Bitte laut mitsingen"

        + LokaleFunktion_LeerZeichenLiefern()      // "Bitte laut mitsingen "

        + LokaleFunktion_DoppelpunktLiefern()      // "Bitte laut mitsingen :"

        + LokaleFunktion_LeerZeichenLiefern()      // "Bitte laut mitsingen : "

        // numerischen Wert 3 konvertieren zu "3"
        + FunktionsLokalesArgument2_numerisch_Wert.toString() // "Bitte laut mitsingen : 3"

        + LokaleFunktion_LeerZeichenLiefern()      // "Bitte laut mitsingen : 3 "

        // "kleine"
}
```




```

+ FunktionsLokaleVariable // "Bitte laut mitsingen : 3 kleine"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen : 3 kleine "

// String-Wert "Neger"
+ FunktionsLokalesArgument2_Wert_String; // "Bitte laut mitsingen : 3 kleine Neger"

GlobaleVariable3 = FunktionsLokaleVariable2; // Inhalt der lokalen Variablen nach
// globale Variable kopieren
// man hätte auch return-Anweisung
// nehmen können
}

// Aufruf mit korrekter Argumentenversorgung durch Parameterliste also
// GlobaleVariable1 "Bitte laut"
// Ausdruck 1 + 2 3
// "Neger"
// Funktion belegt GlobaleVariable3 mit dem Wert "Bitte laut mitsingen : 3 kleine Neger"

Globale_BeispielFunktion(
    GlobaleVariable1, // "Bitte laut"
    1 + 2, // Ausdruck, der den Wert 3 liefert
    "Neger"
);

// Anzeige per Alert-Box (Standard-Funktion) von "Bitte laut mitsingen : 3 kleine Negerlein ..."
alert(
    GlobaleVariable3 // "Bitte laut mitsingen : 3 kleine Neger"

+ GlobaleVariable2 // "Bitte laut mitsingen : 3 kleine Negerlein"

+ GlobaleFunktion_ZeichenLiefern("Blank") // "Bitte laut mitsingen : 3 kleine Negerlein "
// Achtung: Aufruf von LokaleFunktion_DoppelpunktLiefern()
// ist nicht zulässig

+ "..." // "Bitte laut mitsingen : 3 kleine Negerlein ..."
);

```

3.8.3. Funktion vordefiniert

Vordefinierte Funktionen sind Methoden von in der Scriptmaschine definierten Objekten.

Bsp für Methode aller Script-Objekte:

eval() Zeichenkette aus Ziffern, Operatoren, Punkt, Komma und Vorzeichen nach numerisch konvertieren

3.8.4. Funktion durch Programmierer frei deklariert

Die Deklaration erfolgt anhand der Anweisung function oder einer Ableitung vom Script-Objekt Function

function Anweisung:

Deklaration einer frei-programmierten (privaten) Funktion
Verschachtelung möglich z.B. für Rekursion

siehe auch Objekt Function

Syntax:

```

function freier_funktions_bezeichner ([ argumenten_liste ]) // Kopf der Funktion per Anweisung function
{ statements } // Rumpf der Funktion

```

freier_funktions_bezeichner darf kein Schlüsselwort von Script sein

statements Rumpf der Funktion
immer Blockanweisung kodieren
muss nicht return-Anweisung enthalten, kann aber
Hinweis: Eine PROCEDURE gibt es nicht in Script

argumenten_liste Folge von per Komma getrennten Elementen
Listenelement:
immer Referenz
Platzhalter für Parameter (Wert oder Referenz)
Hinweis: intern wird Wert-Parameter auch



referenziert

Wert: z.B. numerisch, Boolean, String, Ausdruck
wird in statements verarbeitet
ist nur funktions-lokal gültig

```
var funktions_name = new Function("argumenten_liste","javascript_anweisungen");
// Ableitung von Script-Objekt Function
```

javascript_anweisungen sind der Funktionsrumpf
in " " bzw. ' ' zu setzen
mit ; trennen

Folgendes Beispiel instanziert keine Funktion als Objekt:

```
function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion();
// bewirkt sofortige Ausführung von TestFunktion() also von alert()
// und liefert keinen Zeiger

// ZeigerAufFunktion(); // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
// da keine Ableitung vom JScript-Objekt Function,
// aber eine Funktion erwartet wird
// Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration
// erkannt
// alert(ZeigerAufFunktion); liefert "(object Object)"
```

Beispiel 1 für Anweisung function:

```
var GlobaleVariable1 = "Bitte laut";
var GlobaleVariable2 = "lein";
var GlobaleVariable3 = "";

function GlobaleFunktion_ZeichenLiefern(ZeichenArt) // ZeichenArt ist funktionslokal
{
    var FunktionsLokaleVariable = "" // Initialisierung des String

    if (ZeichenArt = "Blank")
    { FunktionsLokaleVariable = " ";}

    if (ZeichenArt = "Doppelpunkt")
    { FunktionsLokaleVariable = ":";}

    return FunktionsLokaleVariable; // Funktion liefert Inhalt von FunktionsLokaleVariable
    // und keinen Zeiger auf FunktionsLokaleVariable,
    // da ein Zeiger auf eine funktionslokale Variable
    // niemals lieferbar ist: Zeiger existiert nur zur Laufzeit
    // der Funktion !!
}

function Globale_BeiispielFunktion( // FunktionsLokalesArgument1_Referenz,
// FunktionsLokalesArgument2_Wert_Numerisch
// FunktionsLokalesArgument2_Wert_String
)
{
    function LokaleFunktion_LeerZeichenLiefern()
    { return (GlobaleFunktion_ZeichenLiefern("Blank")); } // liefert " "

    function LokaleFunktion_DoppelpunktLiefern()
    { return GlobaleFunktion_ZeichenLiefern("Doppelpunkt ");} // liefert ":"

    var FunktionsLokaleVariable1 = "kleine";

    var FunktionsLokaleVariable2 =

        // Referenz auf GlobaleVariable1
        FunktionsLokalesArgument1_Referenz // "Bitte laut"

        + LokaleFunktion_LeerZeichenLiefern() // "Bitte laut "
```



```

+ "mitsingen" // "Bitte laut mitsingen"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen "

+ LokaleFunktion_DoppelpunktLiefern() // "Bitte laut mitsingen :"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen : "

// numerischen Wert 3 konvertieren zu "3"
+ FunktionsLokalesArgument2_numerisch_Wert.toString() // "Bitte laut mitsingen : 3"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen : 3 "

// "kleine"
+ FunktionsLokaleVariable // "Bitte laut mitsingen : 3 kleine"

+ LokaleFunktion_LeerZeichenLiefern() // "Bitte laut mitsingen : 3 kleine "

// String-Wert "Neger"
+ FunktionsLokalesArgument2_Wert_String; // "Bitte laut mitsingen : 3 kleine Neger"

GlobaleVariable3 = FunktionsLokaleVariable2; // Inhalt der lokalen Variablen nach
// globale Variable kopieren
// man hätte auch return-Anweisung
// nehmen können

}

// Aufruf mit korrekter Argumentenversorgung durch Parameterliste also
// GlobaleVariable1 "Bitte laut"
// Ausdruck 1 + 2 3
// "Neger"
// Funktion belegt GlobaleVariable3 mit dem Wert "Bitte laut mitsingen : 3 kleine Neger"

Globale_BeispielFunktion(
    GlobaleVariable1, // "Bitte laut"
    1 + 2, // Ausdruck, der den Wert 3 liefert
    "Neger"
);

// Anzeige per Alert-Box (Standard-Funktion) von "Bitte laut mitsingen : 3 kleine Negerlein ..."
alert(
    GlobaleVariable3 // "Bitte laut mitsingen : 3 kleine Neger"

+ GlobaleVariable2 // "Bitte laut mitsingen : 3 kleine Negerlein"

+ GlobaleFunktion_ZeichenLiefern("Blank") // "Bitte laut mitsingen : 3 kleine Negerlein "
// Achtung: Aufruf von LokaleFunktion_DoppelpunktLiefern()
// ist nicht zulässig

+ "..." // "Bitte laut mitsingen : 3 kleine Negerlein ..."
);

```

Beispiel 2 für Ableitung vom Script-Objekt function:

```
var Zeiger = new Function("return init_wert;");
```

Beispiel 3 für Rekursion (Sound mit Sekundenanzeige):

```

<HTML>
<HEAD>
<SCRIPT>
// ++++++ globale Variablen, die verändert werden können
var SoundUrl = "56sec.mid";
var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;

```



```

var ie = document.all ? true : false;

// ++++++++ Routinen der Sekundenzählung
var SekundenZahler      = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen()    // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekundenzählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekundenzählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl      = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet  = true; // kein Sound aktiv
}

// ++++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
}

```



```

    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler            = 0;
    SekundenZahlerTimeoutID   = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdruckes.
    //      Für die Neuberechnung des Ausdruckes ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
                                       "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                       );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                                   + SoundDauerInSekunden.toString()
                                   + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + ' STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

```



```

    );

    document.write(      '<DIV    ID="ID_DIV_SekundenZahler"'
                        +   'STYLE="color:hotpink;font-weight:bold"'
                        + '>'
                        + '</DIV>'
                        + '<BR>'
    );

    document.write(      '<DIV    ID="ID_DIV_MessLatte"'
                        +   'STYLE="color:white;background-color:gray"'
                        + '>'
                        + '</DIV>'
    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ----- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

3.8.5. Funktion als Objektkonstruktor (Objektklasse) per new

Durch die Definierung einer Funktion als Konstruktor, kann ein Objekt erzeugt werden, dessen Eigenschaften und Methoden durch den Konstruktor festgelegt werden, also durch Prototyping. Aber innerhalb der Funktion entfällt die Eigenschaft .prototype, da das Objekt ja beim Instanzieren und **nicht** nachträglich erweitert wird.

siehe new Anweisung

Man beachte, dass eine Funktion selbst als Objekt per new instanziiert werden kann:

Folgendes Beispiel instanziiert keine Funktion als Objekt:

```

function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion();
                        // bewirkt sofortiges Ausführung von TestFunktion() also von alert()

// ZeigerAufFunktion();      // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
                        // da keine Ableitung vom JScript-Objekt Function,
                        // aber ein Funktion erwartet wird
                        // Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration
                        // erkannt
// alert(ZeigerAufFunktion); liefert "(object Object)"

```

Beispiel 1:

```

function erzeuge_zwei_dim_feld(anzahl_spalten, anzahl_zeilen)
{
    // Spaltenfeld erzeugen
    this.spalten_feld= new Array(anzahl_spalten);

    // Zeilenfeld pro Spalte erzeugen
    for (var spalte=0; spalte < anzahl_spalten; spalte++)
    { this[spalte] = new Array(anzahl_zeilen); } //Zeilen-Felder
}
var zwei_dim_tabelle= new erzeuge_zwei_dim_feld(10,7); // 10 Spalten zu je 7 Zeilen
....
zwei_dim_tabelle[10,7]=22; // Spalte 10: 7Zeile: mit 22 belegen

```

Beispiel 2:

```

function Konstruktor()
{

```



```

// objektinterne Rechen-Methode definieren
function Addition(operand1,operand2)
{return (operand1 + operand 2);}

// objektinternes Objekt, das zur Laufzeit erzeugt wird
//      Rechenobjekt definieren, dass nur über die objektteigene Methode erreichbar ist
var internes_rechen_objekt = new Object(); // Script-Objekt mit Eigenschaft .prototype

// objektinternes Objekt erweitern per Prototyping
//      keine Eigenschaften zu definieren, da diese aus Funktionsargumenten operand1 und operand2
//      by Value übernommen werden
//      Methode dem internen Objekt zuweisen
//      Achtung: ohne () kodieren, damit nicht sofort ausgeführt wird
internes_rechen_objekt.prototype.methode= Addition;
}

// Objekt instanzieren
var Zeiger = new Konstruktor();

// Objekt anwenden
alert(zeiger.Addition(10,20)); // zeigt 30 an

```

Beispiel 3:

```
var Zeiger = new Function("return init_wert;");
```

3.8.6. Funktion und Abarbeitungsfolge z.B. bei Rekursion

Grundsätzlich wird ein Dokument zwar sequentiell in der Vorgabefolge des HTML-Interpreters vom Browser gelesen (erst HEAD-Teil, dann der BODY-Teil), aber die Abarbeitung von ausgelösten Aktionen erfolgt immer parallel: Zwei Javascript-Funktionen, die nacheinander im HEAD-Teil des HTML-Dokumenten-Quelltextes kodiert sind, werden nacheinander aufgerufen, arbeiten aber parallel (Multitasking unter Windows).

Der Versuch, durch Schleifen wie for(..) etc. ein Warten der Routinen aufeinander auszulösen, wird mit sofortigem Stillstand des Betriebssystems Windows 9x quittiert, wenn die Schleife z.B. endlos ist. Der Grund ist einfach: In der Schleife finden Zählerveränderungen statt - meist numerische. Diese werden von Windows 9x nicht als Multitasking realisiert sondern als Singletask (Windows 9x kann kein echtes Multitasking). Das ist auch sinnvoll, da Zählerschleifen direkt in Maschinencode und CPU-Register untergebracht werden können, also wenig Ressourcen benötigen.

Die Realisierung von abhängigen Routinen, die z.B. aufeinander warten sollen, ist also nicht durch Warteschleifen möglich sondern nur durch Rekursion. Aber: Rekursionen werden wieder parallel abgearbeitet. Daher ist es nötig, die abhängigen Routinen zu verschachteln oder eine zu den Routinen globale Schaltvariable zu instanzieren, die diese Routinen abfragen. Letztere Variante kann aber nicht garantieren, dass die Routinen nacheinander zugreifen, wenn nicht eine von den beiden Routinen eine längere Wartezeit hat. Diese ist z.B. durch die Methode setTimeout() einrichtbar. Die Methode setInterval() funktioniert wie ein Herzschlag, also endlos bis zum bewussten Abschalten. Die Methode setTimeout() muss immer wieder neu gestartet werden.

Rekursion z.B. per setTimeout() hat einen Nachteil: Es werden die Timer des Betriebssystems benutzt, deren Anzahl begrenzt ist. Je mehr Rekursionen parallel laufen, um so weniger können Betriebssystem-Routinen zugreifen, da diese sich einen Timer teilen müssen. Die Konsequenz: Es kann passieren, dass die Uhr im PC falsch geht, da die Uhr selbst einen immer verfügbaren Timer benötigt. Das Zuweisen von bestimmten Timern ist nicht möglich.

Grundsätzlich sollte für eine Rekursion mit setTimeout() beachtet werden, dass im Programmablauf **nach** Aufruf der Timeout-Anweisung die rekursive Funktion endet und sich somit **unmittelbar** neu startet. Liegen im Programmablauf hinter der Timeout-Anweisung auf **gleicher** Ebene zu ihr noch andere Anweisungen, so wird **mittelbar** rekursiv verfahren: Diese Anweisungen werden **ebenfalls** abgearbeitet, obwohl der Timer bereits aktiv ist oder die Funktion bereits neu gestartet wurde. Wird mit diesen Anweisungen auch noch die Ablaufsteuerung der rekursiven Funktion z.B. in den Bedingungen der Aktivierung des **nächsten** setTimeout() **nachträglich** zur bereits getimten Rekursion beeinflusst, so findet die Funktion bei Rekursion eventuell nicht denjenigen Stand an Daten in Variablen vor, der zum Zeitpunkt des Timerstartes per setTimeout() vorlag und eigentlich erwartet wird. Das kann fatale Folgen für den Programmablauf der Funktion haben: Datenbestände werden also pro Rekursion unerwartet verändert, so dass ein definiertes Ende der letzten Rekursion unvorhersehbar ist. Bei Timerschleifen, die in einem gewissen Zeitraum nicht enden, reagiert entweder der Browser mit einer Empfehlung, dass aktive HTML-Dokument zu beenden, oder Windows stürzt eventuell ab. Als Analogon dient die for-Schleife: Diese Schleifenart wird nicht parallel verarbeitet, sondern zeitecht. Eine for-Schleife, die in ihrer Abarbeitung zeitlich zu lang ist oder nie endet, kann die Performance des Browsers und eventuell des Betriebssystems stark behindern.

Das o.g. Problem tritt bei der Verwendung von setInterval() auch auf, wenn nicht statische Programmabläufe per Zeitintervall aktiviert werden. Es empfiehlt sich, **dynamische** Programmabläufe nicht per setInterval() sondern per setTimeout() zu programmieren, falls möglich. Rekursionen mit setInterval() enden nie, wenn nicht der zugehörige Timer irgendwann gelöscht werden **kann**. setTimeout() verursacht immer einen einmaligen Aufruf, also keine Intervallfolge wie setInterval().

Variablen innerhalb einer rekursiven Funktion sollten **nicht in ihr** deklariert werden, sondern global zur Funktion. Der Vorteil ist, dass die Funktion definitiv immer auf die **selbe** Instanz zugreift. Analog dazu sollte auf Parameter der rekursiven Funktion verzichtet werden, denn damit vereinfacht sich die Adressberechnung zur Rekursion: Eine außerhalb der Funktion deklarierte Variable hat bereits **vor Aufruf** der Funktion eine feste Adresse, so dass der Interpreter des Browsers nur noch diese in die Rekursion einzusetzen braucht und keine



Adressberechnungen durchführen muss. Das erhöht die Browserperformance. Allerdings hat diese Vereinfachung auch einen Haken: Sollte die globale Variable der Funktion dummerweise auch ausserhalb der Funktion angesprochen werden, z.B. weil der Programmierer zwar 2 **verschiedene** Variablen meint, aber diese, ohne es zu bemerken, mit **gemeinsamen** Namen versehen hat, dann kann die Rekursion eventuell fehl schlagen. Daher der Tipp: Variablenbezeichner der rekursiven Funktion immer mit dem Funktionsnamen verbinden, so dass Eineindeutigkeit und damit Übersicht herrschen. Wenn möglich, sind diese Variablen unmittelbar vor dem Funktionskopf zu deklarieren.

3.8.7. Funktion und Rekursionen

3.8.7.1. Übergabe von Variablen an die Rekursion

Mit Start der Rekursion als Argumente an die Routine übergebene Variablen sind routinen-lokal und müssen bei setTimeout() immer wieder neu übergeben werden, d. h., sie werden **pro Aufruf** weitergereicht. Natürlich sind globale Variablen stets innerhalb der Rekursion gültig, also verarbeitbar.

3.8.7.2. Rekursion und document.write() bzw. document.writeln() im HEAD

Eine Rekursion kann anhand von Funktionscode im HEAD realisiert werden.

Beim Internet Explorer ist unbedingt zu beachten:

Die Ausführung von document.write() bzw. document.writeln(), das **HTML-Tags** erzeugt, hat 2 Konsequenzen und zwar genau dann, wenn diese Anweisung **nach dem kompletten Laden des Dokumentes, also nach Auslösung des Ereignisses onload** aktiviert wird:

Fakt 1:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen HTML-Dokumentes**, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

Die Methoden write() und writeln() erzeugen einen Datenstrom aus HTML-Elementen und das neue Dokument empfängt diesen so, als würde er aus einer HTML-Datei stammen. Mit Ende des Datenstromes wird quasi ein Dateiende erkannt und das neue Dokument löst das Ereignis onload aus. Damit wird aber das neue Dokument zum aktuellen Dokument, also im Fenster über dem des alten Dokumentes angezeigt. Da das Fenster des neuen Dokumentes automatisch erzeugt wurde (nicht per Methode open()), sind also die Standards für eine Fenstererzeugung verwendet worden. Damit hat das neue Dokument einen History-Eintrag. Der User kann nun mit diesem zwischen dem alten und neuen Dokument umschalten.

Fakt 2:

Im Falle der o.g. nachträglichen Veränderung des Dokumentes um HTML-Elemente per write() bzw. writeln() kennt das neue, automatisch geöffnete Dokument das alte Dokument nur **als Eltern**. Es muss also im neuen Dokument mit dem Zeiger auf die Eltern gearbeitet werden, wenn Daten und Routinen der Eltern benutzt werden sollen (siehe Objekt window bzw. Objekt document). Mit anderen Worten: Das neue Dokument muss dann komplett per Script erzeugt werden, denn dieser Zeiger lässt sich nur über Script ansprechen. Jedes geladene Dokument hat ansonsten seine eigenständige Umgebung.

Eine Rekursion im alten Dokument, die nach dem Laden des Dokumentes HTML-Tags per Script erzeugt, ist im neuen, automatisch erzeugten Dokument leider nicht bekannt. Man beachte auch: **Jeder** Aufruf der rekursiven Methode erzeugt erneut HTML-Code. Die Rekursion einer Methode sollten also keinen HTML-Code erzeugen, wenn das Dokument, in dem die Rekursion kodiert ist, bereits komplett geparkt wurde.

3.1.9. ASCII-Code und Unicode

ASCII: pro Zeichen 7 Bit verwendet, also 128 Zeichen möglich (0-127)
enthalten im Unicode

Eurozeichen ab HTML4.0 kodierbar per € oder AC oder &euro

Unicode: pro Zeichen 16 Bit verwendet, also 65535 Zeichen möglich
Zeichen 0-127 identisch mit ASCII-Code

Kodierung als ESC-Sequenz \uxxxx mit xxxx als Hexaziffern

Beispiele: \u0008

Rückschritt

\u0009	horizontaler Tabulator
\u000A	Zeilenvorschub (Line Feed) neue Zeile
\u000B	vertikaler Tabulator
\u000C	Seitenvorschub (Form Feed)
\u000D	für Wagenrücklauf
\u0020	Blank, Leerzeichen
\u0022	"
\u0027	'
\u005C	\

Spezialzeichen in Microsoft JScript



Escape Sequenzen

\b	Backspace, Rückschritt
\f	Form feed, Blattvorschub
\n	Line feed (newline), Zeilenvorschub
\r	Carriage return , Wagenrücklauf (nicht Enter !!)
\t	Horizontal tab (Ctrl-I), horizontaler Tabulator
\'	Entwertung des Zeichen '
\"	Entwertung des Zeichen "
\\	Backslash z.B. für lokale Pfadangabe wie C:\\test\\bilder\\test.gif.
\\\\	Entwertung Backslash
\\xhh	hexadezimal aber nur von 00 bis FF
\\uhhhh	Unicode

Beispiele:	\\u0008	Rückschritt
	\\u0009	horizontaler Tabulator
	\\u000A	Zeilenvorschub (Line Feed) neue Zeile
	\\u000B	vertikaler Tabulator
	\\u000C	Seitenvorschub (Form Feed)
	\\u000D	für Wagenrücklauf
	\\u0020	Blank, Leerzeichen
	\\u0022	"
	\\u0027	'
	\\u005C	\\

nicht druckbare Zeichen

z.B. siehe Machcode der Objekte regexp bzw. RegExp

3.10. Fehlerbehandlung in Javascript

Es ist darauf zu achten, dass der Browser die HTML-Dokumente nicht aus dem Browser-Cache liest, solange die Dokumente nicht fehlerfrei laufen, also beim Test der Dokumente (Daten im Browser-Cache vor jedem Test löschen).

Das generelle Lesen aus dem Cache ist per Script, META-Tag, sowie per Browsereinstellung abänderbar. Letztere kann der User treffen, in dem er z.B. beim Internet Explorer die Cache-Löschung mit Schließen des Browser abhakt.

Es ist üblich, dass der User den Browser-Cache **nicht löscht**, da der Browser den Cache-Füllstand automatisch verwalten kann. Daher muss der Programmierer damit rechnen, dass die HTML-Dokumente aus dem Cache geladen werden könnten. Deshalb ist es wichtig, dass der Programmierer der Webseiten dafür sorgt, dass per META-Tag immer ein Datum der Webseite besteht, das mit Sicherheit verfallen ist und somit die HTML-Daten vom Server geladen werden. Achtung: Des erneute Laden einer Webseite (Reload), z.B. innerhalb eines bestimmten Zeitraumes, sollte aus dem Cache kommen, damit nicht Daten doppelt zu laden sind, wenn sie bereits gültig im Cache liegen. Will der Programmierer absolut sichergehen, dass nach einem bestimmten Zeitraum die Daten vom Server zu laden sind **und** soll der User dieses Verhalten nicht beeinflussen können, dann muss das Verhalten per Script programmiert werden (Algorithmus ist Sache des Programmierers). Zugleich wird der User und der Server mehr Traffic beim Laden der Daten der HTML-Dokumente haben.

Fehlermeldung beim Internet Explorer zu Objekten:

z.B. "Objekt nicht gefunden" oder "Objekt ... unterstützt Eigenschaft nicht" oder "CLASS nicht gefunden"

Die Fehler, die per Fehlermeldungen zu Objekten im Internet Explorer angezeigt werden, können dessen im Speicher instanzierte Objektstruktur durcheinander bringen, so dass nach Abhilfe des Fehlers im Quelltext und anschließender Aktualisierung des Dokumentes (per Aktualisierungs-Button im Browsermenü) trotzdem dieselbe Fehlermeldung erscheint. Es ist daher ratsam, bei solchen Fehlermeldungen den Browser zu schließen, den Browser-Cache zu löschen und dann den Browser mit dem Quelltext neu zu starten. Ursache ist ein eventuelles automatisches aber fehlerhaftes Prototyping von Objekten (auch von browserinternen). Hinweis: Man sollte immer mit genau 1 Instanz des Browsers testen, damit Objektstrukturen konsistent bleiben können.

Fehlermeldungen zum Script

z.B. "Scriptfehler"

Fehlermeldung kann echten Fehler anzeigen (z.B. wegen falscher Browserversion) **oder** Speichermangel, welcher die Abarbeitung des Skriptes verhindert (nicht mögliche Instanzierung von Objekten, Variablen etc.). Bei Speichermangel kann der Neustart des Browsers helfen.

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

Bsp.: var Kette = 'test'; + '.jpg'; // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen +
 alert(Kette); // zeigt nur test an und **nicht** test.jpg

Bsp: for (var i = 0; i <3; i++); // **kein** Syntaxerror wegen dem Semikolon vor dem Zeichen {
 {alert(i);} // genau 1 Anzeige, die 3 anzeigt



3.10.1. Runtime Fehler (Laufzeitfehler) von JScript (Auswahl)

siehe auch error JScript-Objekt des Internet Explorer für private Run-Time-Error

5000	Zeiger laut this Anweisung nicht zuweisbar
5001	Instanz vom Objekt Number erwartet
5002	Instanz von Objekt Function erwartet
5003	Funktionsergebnis nicht zuweisbar
5005	Instanz von Objekt String erwartet
5006	Instanz von Objekt Date erwartet
5007	Instanz (Objekt) erwartet
5008	unzulässige Zuweisung
5009	undefinierter Bezeichner
5010	Instanz von Objekt Boolean wird erwartet
5012	Instanz vom Object member erwartet
5013	Instanz von Objekt VBArray erwartet
5014	Instanz eines JScript-gültigen Objektes erwartet
5015	Instanz von Objekt Enumerator erwartet
5016	Instanz von Objekt Regular Expression erwartet
5017	Syntax Fehler im regulären Ausdruck
5018	undefinierte Qualifizierung (Punktnotation)
5019	'[' erwartet im regulären Ausdruck
5020	']' erwartet im regulären Ausdruck
5021	falscher Mengenbereich im Character Set
5022	Ausnahme per thrown Anweisung erwartet
5023	Funktion hat kein zulässiges Prototyping (Funktionskopf falsch)
5024	URI enthält fehlerhaftes Zeichen
5025	URI ist ungültig für encoding
5026	Anzahl der Ziffern hinter dem Komma ist ungültig
5027	Bereichsgrenze überschritten
5028	Instanz von Objekt Array oder Objekt arguments erwartet
5029	Feldlänge ist nicht >= 0 nicht Integer nicht endlich
5030	Feldlänge ist nicht >= 0 nicht Integer

3.10.2. Syntax-Fehler von JScript (Auswahl)

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser leider **nicht als fehlerhaft** erkannt werden:

```
Bsp.: var Kette = 'test'; + '.jpg'; // kein Syntaxerror wegen dem Semikolon vor dem Zeichen +
      alert(Kette); // zeigt nur test an und nicht test.jpg

Bsp:  for (var i = 0; i <3; i++); // kein Syntaxerror wegen dem Semikolon vor dem Zeichen {
      {alert(i);} // genau 1 Anzeige, die 3 anzeigt
```

Es gibt Konstellationen im Quellcode des Skriptes, die vom Browser als irritierend erkannt werden:

```
Bsp: anstelle von if wird if (also if mit Hütchen) kodiert --> der Browser verlangt ein Objekt und erzeugt keinen
      üblichen Syntaxerror zur if-Anweisung
```

Es wird dringend angeraten, einen JavaScript-Debugger zu verwenden, vorallem dann, wenn die Quelltexte gross sind (debuggen = entwanzen von Fehlern).

Debuggersoftware ist in der Regel Bestandteil eines Design-Paketes und für Hobby-Programmierer nicht gerade preiswert. Wichtig: Im Browser müssen Script-Debugging und die Anzeige von Script-Fehlermeldungen genehmigt sein (Nachteil: Bei einer Onlinesitzung wird sich der User bei diesen Einstellungen wundern, wie oft Script-Fehler auch auf (bei abgeschaltetem Scriptdebugging als) professionell wirkenden Seiten auftauchen (z.B. weil nicht browsergerecht programmiert wurde oder über Link eingebundener Scriptcode benötigt wird, der z.B. auf einem anderen, aber gerade nicht erreichbaren Server liegt) und daher eventuell eine Debugging-Anfrage(-Folge) erzeugen.

Einen **kostenlosen** Script-Debugger bietet Microsoft an, der seit Jahren nicht mehr gepflegt wird, vom Design und Umfang abenteuerlich primitiv (steinzeitlich) ist und vermutlich deswegen als kostenlose Software (Gnadenbrot für Hobby-Programmierer) bis in die Ära der hypermodernen, kommerziellen Windows-Net-Software überlebt hat, aber zumindest Script-Fehler mehr oder weniger **erkennt**. Nachfolgend weitere "Vor-" und Nachteile:

Trotz Zeilenangabe in der Fehlermeldung positioniert der Debugger in der Regel auf die Zeile, die in der Anzahl der Zeilen im Dokument genau in der Mitte liegt und nicht auf die fehlerhafte Zeile.

Der Debugger ist so alt, dass er die Maus-Rad-Bedienung nicht kennt (Scrollen per Maus nicht möglich).

Der Debugger kann nicht mit Framesets umgehen: Es wird **immer** nur das Dokument mit dem FRAMESET-Tag angezeigt (natürlich dann auch die, wie oben genannt, falsche Zeilenposition), auch wenn der Fehler in einem Frame auftritt. Frameseiten sind also erst solo auf Fehler zu testen und dann - wagemutig - im Frameset.

Der Quelltext ist nur dann im Debugger schreibbar, wenn er nicht parallel offen ist. Mit anderen Worten: Wer einen anderen Editor parallel benutzt, hat Pech. Der Debugger besitzt eigene Einstellungen zu Tabs etc..



Der Debugger lässt sich manchmal manuell nicht über den Menüpunkt Ansicht öffnen, wenn der Debugger trotz Fehlermeldung und dortigem geklickten Debuggerstart sich einfach nicht öffnet (aber die Fehlermeldung natürlich verschwindet). Dann ist oft Rätselraten angesagt, wenn der **nun doch** parallel zu benutzende Editor (nur Plain-Text, also z.B. notepad.exe und nicht wordpad.exe) keine Zeilenpositionierung kennt. bzw. diese bei grossen Dateien falsch ausführt (z.B. Editor im Symantec Norton Commander der Windows-Version).

Tritt ein Fehler auf, so **muss** in der Regel der Debugger geöffnet werden (immer dann, wenn der Fehler von der Scriptmaschine nicht übergebar ist), bevor die Webseite weiter im Browserfenster begutachtet werden kann und das zusätzlich oft nur **nach** dem Schliessen des Debuggers bzw. Quelltextes im Debugger.

Bei Syntaxfehler wegen vergessener } oder) bzw. { oder (ist sehr oft der Debugger nicht in der Lage, die Stelle des Fehlers anzuzeigen. Grund: Die Struktur der Klammerung im gesamten Dokument wird solange geparkt, bis eine Klammerung definitiv als fehlend erkannt wird, das aber an einer Stelle, die nicht der Position entsprechen muss, an der die Klammer vom Programmierer wirklich vergessen wurde. Dann ist im wahren Sinne des Wortes großes und zeitraubendes Rätselraten angesagt, um die echte Fehlerstelle zu finden. Deswegen: Konzentriert und sauber programmieren (auf Einrückungen bei Verschachtelung unbedingt achten, Kommentare im Quellcode erzeugen, Schnittstellen der Datenübergabe bilden etc.).

Programmierungssoftware, die selbst JavaScript innerhalb der Software für HTML-Design nutzt, ist zum Debuggen nicht zu empfehlen (falls im Design-Paket überhaupt ein Debugger vorhanden ist). Das Problem: Per JavaScript unterstütztes HTML-Design (z.B. visuelle Feinausrichtung eines HTML-Elementes per Maus oder vorgefertigte Menüs) kann JavaScript-Code des Herstellers einbauen. Dieser Code wird natürlich auch debuggt und erschwert die Situation des Programmierers vor allem dann, wenn dieser Code mit programmierer-eigenem Scriptcode nicht über vom Hersteller vordefinierte Schnittstellen (falls diese überhaupt vorhanden und kommentiert sind) erweitert wird und deswegen zusätzlich nicht lauffähig sein kann.

Die Qualität eines Quelltextes entscheidet auch über Fehleranfälligkeit

Beispiele für schlechte Qualität aus eindeutiger Faulheit: Manche Programmierer halten sich mit ihrem Produkt für so perfekt, dass sie nicht an denjenigen denken wollen, der den Quelltext verstehen muss: Die Kreativität im Quelltexterzeugen ist eine Intelligenzfrage und nicht die der Selbstbefriedigung. Letztere Intention ist massenhaft im Internet zu finden, so dass die Quelltextanalyse wohl bedacht sein will (vor allem um dann festzustellen, dass der Programmierer im Wahn seiner Faulheit dein eigenen Quelltext nicht versteht und Fehler eingebaut hat).

Zur Beruhigung: Im professionellen Bereich finden sich ebenfalls Faule, die dann Programmierer, die ihre Quelltexte so strukturieren, dass Dritte sie lesen und verstehen können, als Literaten bezeichnen. Niemand ist unersetzbar, besonders Faule als Angestellte im Job, deren Faulheit im geistigen Eigentum des Arbeitgebers endet, wofür der Arbeitgeber auch noch zahlt. (Sehr beliebt ist das bei C-Programmierern, die ihren Quellcode zum Stacheldraht-Verhau umwandeln und Logik mit möglichen wenigsten Quellcode implementieren (und dabei auf Compiler-Intelligenz setzen)

Dass im Quelltext keine Kommentare enthalten sind, sei aus Vereinfachung nicht weiter beachtet.

Bsp. 1

```
<script language="JavaScript">

var NS4 = (document.layers);
var IE4 = (document.all);

var win = window;
var n = 0;

function findInPage(str) {

    var txt, i, found;

    if (str == "")
        return false;

    if (NS4) {

        if (!win.find(str))
            while(win.find(str, false, true))
                n++;
        else
            n++;

        if (n == 0)
            alert("Nichts gefunden.");
    }

    if (IE4) {
```



```

txt = win.document.body.createTextRange();

for (i = 0; i <= n && (found = txt.findText(str)) != false; i++) {
    txt.moveStart("character", 1);
    txt.moveEnd("textedit");
}

if (found) {
    txt.moveStart("character", -1);
    txt.findText(str);
    txt.select();
    txt.scrollIntoView();
    n++;
}

else {
    if (n > 0) {
        n = 0;
        findInPage(str);
    }

    else
        alert("Nichts gefunden.");
}

return false;
}

</script>

<form name="search" onSubmit="return findInPage(this.string.value);">
<font size=3><input name="string" type="text" size=15 onChange="n = 0;"></font>
<input type="submit" value="Suchen">
</form>

```

Jetzt den maschinell aufbereiteten Quellcode, an dem man sehen kann, wie sinnlos o.g. Faulheit ist. (Wer allerdings die Aufbereitung manuell machen muss, dem ist nur zu gratulieren :-))

```

<script language="JavaScript">
var NS4=(document.layers);
var IE4=(document.all);
var win=window;
var n=0;

function findInPage(str)
{
    var txt,i,found;
    if(str=="")
        return false;
    if(NS4)
    {
        if(!win.find(str))
            while(win.find(str,false,true))
                n++;
        else
            n++;
        if(n==0)
            alert("Nichts gefunden.");
    }
    if(IE4)
    {
        txt=win.document.body.createTextRange();
        for(i=0;i<=n&&(found=txt.findText(str))!=false;i++)
        {
            txt.moveStart("character",1);
            txt.moveEnd("textedit");
        }
        if(found)
        {

```



```

        txt.moveStart("character",-1);
        txt.findText(str);
        txt.select();
        txt.scrollIntoView();
        n++;
    }
    else
    {
        if(n>0)
        {
            n=0;
            findInPage(str);
        }
        else
            alert("Nichts gefunden.");
    }
}
return false;
}
</script>
<form name="search" onSubmit="return findInPage(this.string.value);">
<font size=3><input name="string" type="text" size=15 onChange="n = 0;"></font>
<input type="submit" value="Suchen">
</form>

```

Bsp. 2:

```

<BODY>
<a href="javascript:function C(v){return '<td>'
+v+'</td><td>'+((v>>4).toString(16)+(v&15).toString(16)).toUpperCase()+'</td><td bgcolor=DDDDDD><b>&'+'#
+v+'</b></td>';} var c=4,b=Math.ceil(224/c),a='<table border=0><tr>';for(j=0;j<c;j++){
a+='<td>DEC</td><td>HEX</td><td><b>ASC</b></td>';}a+='</tr>';for(i=33;i<33+b;i++){a+='<tr>';
for(j=0;j<c;j++){t=i+(j*b);if(t<=255)a+=C(t);}a+='</tr>';}a+='</table>';var W=open("",'width=500,height=600,
left=0,top=0,resizable,scrollbars');W.document.writeln(a);">Klick</a>
<BODY>

```

hier der maschinell aufbereitete Quellcode

```

<BODY>
<a href="javascript: function C(v)
{
    return
    '<td>'
    +v+'</td><td>'
    +((v>>4).toString(16)+(v&15).toString(16)).toUpperCase()
    +'</td><td bgcolor=DDDDDD><b>&'
    +'#'
    +v
    +'</b></td>';
}
var c=4,b=Math.ceil(224/c),a='<table border=0><tr>';
for(j=0;j<c;j++)
{
    a+='<td>DEC</td><td>HEX</td><td><b>ASC</b></td>';
}
a+='</tr>';
for(i=33;i<33+b;i++)
{
    a+='<tr>';
    for(j=0;j<c;j++)
    {
        t=i+(j*b);if(t<=255)a+=C(t);
    }
    a+='</tr>';
}
a+='</table>';
var W=open("",'width=500,height=600,left=0,top=0,resizable,scrollbars');
W.document.writeln(a);">Klick</a>
<BODY>

```



Die Qualität eines Quelltextes entscheidet auch über Schnelligkeit des Parsers:

Je einfacher strukturiert der Quelltext ist, um so schneller ist der Parser.

z.B. benötigt eine case-Anweisung mehr Parser-Aufwand als eine Folge von optimal verschachtelten ifs.

z.B. bewirkt die Zwischenspeicherung eines Zeiger, auf den mehrmals zugegriffen werden soll, eine Vereinfachung für den Parser, da der Zeiger nur 1x berechnet werden muss.

z.B. bewirkt die Nutzung von globalen Variablen in einer Rekursion einen Geschwindigkeitsvorteil, da lokale Variablen nicht pro Rekursion zu erzeugen sind.

Nachfolgend ein Beispiel für den Parser schlecht optimierten Quellcode: Test der Cookiverwaltung im Browser.

```
<script language="JavaScript">

function getCookieVal(offset)
{
  var endstr=document.cookie.indexOf(";",offset);
  if(endstr==-1)
    endstr=document.cookie.length;
  return unescape(document.cookie.substring(offset,endstr));
}

function GetCookie(name)
{
  var arg=name+"=";
  var alen=arg.length;
  var clen=document.cookie.length;
  var i=0;
  while(i<clen)
  {
    var j=i+alen;
    if(document.cookie.substring(i,j)==arg)
      return getCookieVal(j);
    i=document.cookie.indexOf(" ",i)+1;
    if(i==0)break;
  }
  return null;
}

function SetCookie(name,value)
{
  var argv=SetCookie.arguments;
  var argc=SetCookie.arguments.length;
  var expires=(argc>2)?argv[2]:null;
  var path=(argc>3)?argv[3]:null;
  var domain=(argc>4)?argv[4]:null;
  var secure=(argc>5)?argv[5]:false;
  document.cookie=name+"="+escape(value)+
    ((expires==null)?"":("; expires="+expires.toGMTString()))+
    ((path==null)?"":("; path="+path))+
    ((domain==null)?"":("; domain="+domain))+
    ((secure==true)?"; secure":"" );
  }
  var enabled="";
  var exp=new Date();
  exp.setTime(exp.getTime()+(60*1000));
  SetCookie('wannasomcookie',1,exp);
  enabled=GetCookie('wannasomcookie');
</script>
<script language="JavaScript">
if(enabled==null)document.write("<b><center>Ihr Browser akzeptiert keine Cookies<br>Your Browser does not accept
Cookies");
else document.write("<b><center>Ihr Browser akzeptiert Cookies<br>Your Browser does accept Cookies");
</script>
```

Fehleranalyse im Internet Explorer

Die Fehleranalyse des IE ist das unterste Niveau, was es im Bereich Programmierung auf Parser-Basis (ohne Compiler) und ohne Hilfesystem per Fremdtool gibt. Die Fehleranalyse des IE liegt im Niveau unterhalb jedes anderen Basic-Sprache-Systems z.B. aus DOS-Zeiten, oder unterhalb diverser Freeware-Parser.

Die Fehleranalyse kann sehr stark eingeschränkt werden, wenn am Quelltextanfang kodiert wurde:



```
function SymError(){return true;}window.onerror = SymError;
```

Der Funktionsname ist beliebig.

Genau dann unterbleiben oft Fehlererkennungen.

Wer mit o.g. Funktion das Dokument testet, testet umsonst !

Die Funktion sollte eigentlich nie kodiert werden, es sei denn, man will Scriptfehler verheimlichen, weil das Dokument nicht ausreichend getestet wurde bzw. man mit Scriptfehlern bewusst rechnet.

Der Parser des IE ist ausschliesslich auch Abarbeitung des Dokumentes orientiert und unterstützt Fehlerbehandlung NUR aus dieser Sicht: Fehlermeldungen kommen NUR mit Daten, die der Parser aus seiner Sicht liefert, die aber nicht der des Programmierers entspricht (Beispiele siehe unten bezüglich alert()-Verwendung). Mit anderen Worten: Das Parsen ist Bedingung für Laufzeit des Dokumentes. Fehler, die der Parser anzeigt, verhindern das Laufen des Dokumentes. Und darauf macht der Parser aufmerksam. Nur wegen Ausführung des Dokumentes kommen Fehlermeldungen (Beispiel: siehe unten).

Wer es schafft herauszubekommen, wie der Parser des IE im Einzelnen arbeitet, kann sich viel Zeit ersparen (Parserversionen je nach Version des Windows Script Host also auch des Betriebesystemes und/oder Browserversionen).

Ohne Wissen über den Parser (z.B. aus reichlich Testerfahrung) gilt:

Wer mit dem IE ohne Fremdtools programmieren will, muss sich auf reichlich Ärger und Minimalsupport durch den IE einstellen, oder gleich die Finger davon lassen.

Dann hilt nur entweder Fremdtools zu nutzen oder manuell den gesamten Quelltext der JS-Datei nach Fehler zu durchforsten (Methoden siehe unten). (Es können aus selbstprogrammierte Tools das Manko ersetzen.)

Grössere JS-Dateien oder Abschnitte des Dokumentes müssen daher schon in Teilabschnitten der Kodierung getestet werden (Testphase parallel zur Kodierung im Try-And-Error-Stil).

Beispiel: Dieses Beispiel verursacht einen Fehler, der eine fehlende ')' annimmt
Zeile 5 Zeichen 14 also an der Stelle wo '!' steht.

```
123456789012345678901234567890
```

```
-----
1      <html>
2      <head>
3      <SCRIPT LANGUAGE='JScript'>
4      <!--
5      var X10=(X04 ! null);
6      //-->
7      </SCRIPT>
8      </head>
9      </html>
```

'!' ist der NOT-Operator, der eine nachfolgende Referenz erwartet;
im Code wäre das also die Referenz null.

Da null ein Zeigerwert ist (Bsp. var x=new Array(); alert(X!=null); liefert true)
kann er nicht negiert werden.
Also ist ! falsch.

Weil der Operator falsch kodiert wurde, muss der Parser den vorhergehenden
gültigen Code für komplett gültig machen, indem die Klammer
vorgeschlagen wird, was aber aus Programmierersicht völliger
Unsinn ist, weil nur '=' vergessen wurde zu kodieren.

Der Parser will also
var X10=(X04)
was syntaktisch korrekt ist.

Kodiert man die Klammer, dann ergibt sich:

```
var X10=(X04) ! null;
oder var X10=(X04) null;
```

Ergo nächster Fehler: Der Parser erwartet ein ';



... und so weiter

Der Parser prüft also genau bis zur Fehlerstelle 14 in Zeile 5,
aber prüft nicht, welche Kombinationen es mit dem als
falsch-kodiert-erkannten Operator '!' gibt
z.B. !=
und bemerkt daher nicht ein '=' oder anderes Relationszeichen.
(Der Hinweis "Relationszeichen" neben dem Hinweis auf ')' würde schon ausreichen).

Mit anderen Worten: Die Kontextanalyse des IE ist minimalistisch und
häufig wertlos. Wieso Microsoft den Parser so
schlecht als Schnittstelle ausstattet, ist völlig unklar.

Jedes Entwicklungssystem, das NUR auf Informationen des Parsers zurückgreift, kann nur so gut sein wie der Parser.

Fremdtools können im Regelfall nicht so optimiert sein wie Tools des Browserherstellers, es sei denn,
der Browserhersteller legt Internas komplett offen, was sehr stark zu bezweifeln ist.
implementieren oft eigenen Scriptcode als Feature des Tools.

Analyse-Methode:

Es muss unbedingt auf Kommentierung des Quelltextes geachtet werden, da Kommentare
ein Hilfe darstellen können.

Der Browser parst schrittweise die Bereiche zwischen <script> und </script>
in der Reihenfolge der kodierten <script> und </script>.

Eine Fehlermeldung bezieht sich also immer auf einen Bereich zwischen <script> und </script>.

Die Frage ist nur, welchen Bereich zwischen <script> und </script>, wenn mehrere
<script> und </script> also Bereiche vorliegen.

Um festzustellen WELCHE Fehlermeldungen bis zum o.g. </script> generiert werden,
muss ein <script>alert();</script>
hinter obigem </script> kodiert werden.

Es sind dann ALLE Scriptfehler VOR der alert-Meldung zu beheben,
auch wenn nach der alert()-Meldung weitere Scriptfehler angezeigt werden,
die aber aus anderen <script> und </script> stammen können.

Sollte ein alert() am Anfang des Quelltextes nicht erkannt werden, so ist im gesamten
Quelltext ein Scriptfehler:

Der Parser prüft z.B. Paarigkeit von Klammern: Fehlt z.B. genau 1 Klammer
im Quelltext um Funktionen, dann ordnet der Parser die nächste
gefundene Klammer zu, bis das Ende des Quelltextes erreicht wird. Und
genau an dieser Stelle tritt der Scriptfehler auf mit der Nummer der
letzten Zeile im Quelltext, obwohl die fehlende Klammer irgendwo im
Quelltext fehlt. Dann muss der gesamte Quelltext manuell geprüft werden
auf Syntaxfehler.

Wenn der Parser einen Scriptfehler wegen fehlender Klammer inmitten des
Quelltextes anzeigt und auffordert, dass eine Klammer an die Position
laut Fehlermeldung zu setzen ist, dann kann nach Setzung der Klammer
ein erneuter Scriptfehler auftreten: Dann verlangt der Parser zwar
eine Klammer, aber nicht an der Stelle. Also muss die Klammer
woanders fehlen, oder die Klammer ist wegen anderem Syntaxfehler,
der erst später erkannt wird, garnicht notwendig. Wie auch immer:
Dann muss der gesamte Quelltext manuell geprüft werden
auf Syntaxfehler.

Beispiel: Dieses Beispiel verursacht einen Fehler, der eine fehlende '}' annimmt
Zeile 18 Zeichen 1 also vor //-->

123456789012345678901234567890

```
-----
1      <html>
2      <head>
3      <SCRIPT LANGUAGE=JScript>
4      <!--
```



```

5      function z()
6      {
7          if (X10)
8          {
9              if (X08==0)
10             {
11                 X10=(X01 != null);
12                 if (X10){X11=X01.length;}
13                 if (X10){X10=(X02!=null);
14                 if (X10){X12=X02.length;}
15             }
16         }
17     }
18     //-->
19     </SCRIPT>
20     </head>
21     </html>

```

Der Fehler liegt in Zeile 13 am Zeilenende: Dort fehlt '}'.

Der Parser ordnet also anstelle der fehlenden Klammer nichts zu, sondern verändert die Programmlogik:

```

function z()
{
    if (X10)
    {
        if (X08==0)
        {
            X10=(X01 != null);
            if (X10){X11=X01.length;}
            if (X10){
                X10=(X02!=null);
                if (X10){X12=X02.length;}
            }
        }
    }
}

```

fehlende Klammer laut Parser: Am Ende der Funktion, das zufälligerweise vor dem //--> steht

Der obige fehlerhafte Code inmitten eines Quelltextes mit diversen Funktionen und Verschachtelungen verursacht Stress pur.

Besonders stressig wird folgender Fall:

alle Inklude sind eigenständig syntaktisch korrekt
im Dokument das inkludiert wird ein '}'-Klammer-Fehler entdeckt
jedoch ein alert hinter dem letzten Inklude wird erkannt
aber nicht das alert hinter <script> das direkt hinter dem letzten Inklude
liegt (nach Tagwechsel) und VOR der ersten Klammer '{'
im Dokumentenquelltext

Damit können doch die Inklude nicht syntaktisch korrekt sein !

Da Inklude als Quelltext eingebunden werden wird eine Klammeranalyse auch über die Include vollzogen, also die Programmierlogik über eventuell alle Include verändert, obwohl die Inkludes einzeln korrekt sind.

Nur in welcher Inklude liegt der Fehler ? Also ab welcher Inklude wird die Programmlogik verändert ?

Also alle Inklude als Gesamtbestand syntaktisch testen:
Quelltexte aller Include in neues Dokument kopieren und das testen.

Wird dann ein Fehler angezeigt, der im Dokument auf ein HTML-Tag weist, das nichts mit Script zu tun hat, z.B. auf <head>, aber dort ist kein Fehler, dann ist klar, den Parser kann man endgültig vergessen.

Stunde der Wahrheit: Elan ohne Parser weiter zu machen oder sich andere (kostenpflichtige) Software



- ev. inklusive anderen Browser-
beschaffen.

Eines steht fest: Viel Spass bei der Try-And-Error-Programmierung :-)

Wird beim Rendern Script-Quelltext im Browserfenster angezeigt, dann wurde ein <script>-Tag
oder </script>-Tag nicht korrekt erkannt.

Der im Browser angezeigte Quelltext könnte zur Suche im Script helfen, muss
aber nicht: Die Anzeige muss nicht Text anzeigen, den es so - wie angezeigt -
im Script tatsächlich auch gibt.

Wegen Fehler könnten Scriptbereiche virtuell in der Anzeige zusammengeschoben sein.
Mann muss dann nach eindeutigen Stellen suchen, die identisch sind in der
Anzeige und im Script.

Achtung: Kommentarzeichen // werden ebenfalls so behandelt wie bei Klammeranalyse

Beispiel: // text document.write('<SCRIPT LANGUAGE="JScript" SRC="...."></SCRIPT>');

Diese Zeile wird als fehlerhaft erkannt, wenn das // nicht korrekt erkannt
wurde und dadurch document.write(....) nicht hinpasst aus Sicht
des Parsers.

Entfernt man diese Zeile und es wird kein Script mehr in der Anzeige gerendert,
MUSS ein //-Fehler vorliegen ! ... Nur wo ?????

Hier hilft alert(); , denn wird das ausgeführt, dann ist
Script auch geparkt worden und gültig.

alles Script, was nach alert(); im Browserfenster als Text angezeigt wird,
ist falsch.

Man beachte: alert() kann wegen //- oder Klammerfehler-Programmlogik-

Verschiebung

aktiv werden !

Ist die Fehlerstelle aber korrekt, so liegt eine Programmlogik-Verschiebung vor.
Also muss der Script vor dem alert() untersucht werden per
stückchenweiser Quelltextentfernung. Und zwar solange, bis
Script nicht mehr als Text NACH alert(); angezeigt wird.
Dann hat man die Fehlerstelle, die VOR alert() liegt.

Folgendes Beispiel rendert Script als Text mit folgender Anzeige '); //-->

```

1      <html>
2      <head>
3      <SCRIPT LANGUAGE='JScript'>
4      <!--
5      // Das ist ein Text --> siehe nachfolgendes alert();
6      alert();
7      // Das ist auch ein Text: So bindet man Script dynamisch ein:

document.write('<SCRIPT

// LANGUAGE="JScript" SRC="...."></SCRIPT>');
8      //-->
9      </SCRIPT>
10     </head>
11     </html>
```

Der Parser macht folgendes draus:

');//--> deutet auf Scriptfehler VOR dem ' hin aus Zeile 7
Das letzte > in Zeile 7 wird erkannt, da nicht angezeigt.

Aber mal Zeile 5 ansehen !! dort befindet sich ebenfalls ein '>' und könnte
einen TAG markieren

Wird dieses '>' in Zeile 5 entfernt, ist der Fehler weg !!!

Werden in Zeile 5 direkt VOR dem '>' die beiden '-' entfernt, ist
der Fehler auch weg.

'-->' wird also vom Parser erkannt.

alert in Zeile 6 wird immer ausgeführt, also ist Zeile 6 okay.

kann nur ausgeführt werden, wenn Zeile

3



oaky ist.

Also liegt der Fehler in den Zeilen 4 bis 5.

wegen '--> in Zeile 5 wird '>' am Ende der Zeile 7 erkannt

wobei '--> aus Zeile 8 nicht als

Tagbegrenzer erkannt wird.

Wer mag, kann austesten wie der Browser reagiert, nimmt man anstelle

Zeile

5 nun in Zeile 7 ein '>' weg.

Als letztes Mittel hilft, Quelltextteile stückweise solange zu entfernen, bis kein Scriptfehler mehr auftritt, so dass der Fehler dann im entfernten Quelltext liegen muss.

Diesen entfernten Quelltext kopiert man dann in ein leeres Dokument und testet mit diesem wie oben beschrieben.

Fehler, die zur Laufzeit erzeugt werden z.B. Variable nicht definiert, werden dann Stress, wenn z.B. mehrere Inklude vorhanden sind, die als Bezeichner für lokale Variablen (innerhalb einer Funktion) identische Bezeichner verwenden. Dann ist unklar, welche Inklude, da die Fehlermeldung auf das Dokument bezogen sein kann, auch wenn Fehler in einem Inklude liegt. Als Orientierung kann die Rendering des Dokumentes verwendet werden: Der Programmierer weiss, was welche Include bewirkt. Desweiteren müssen Programmaufrufe vom/aus dem Inklude geprüft werden.

Typisches Beispiel:

Parameter einer Funktion sind in der Argumentenliste lokal referenziert. Daher können alle Argumente mit dem Bezeichner über alle Funktionen identisch sein.

Zulässig ist z.B.

```
function y(X00,X01,X02)
{ ... }
```

```
function z(X00,X01,X02)
{ ... }
```

Scriptfehler können einen Absturz des Browsers verursachen, ohne dass der Absturz sichtbar wird:

Es ist möglich, dass das DOM zerschossen wird und nur ein Neustart des Browsers hilft.

Grund: Der Übergang zwischen Parsen und Ausführen des Scriptes ist nicht sichtbar.

Wenn ein Script zwar syntaktisch korrekt, aber z.B. wegen Zeigerfehler das DOM zerschiesst, so kann der Parser nicht mehr korrekt arbeiten, was auf die Fehleranalyse Auswirkungen hat (z.B. unter unbemerkt zerschossenem DOM den Quelltext ändern und dann neu parsen lassen).

Meist ist dann vergebliche z.T. erhebliche Scripttestzeit vergangen, eh man den Absturz bemerkt. Lieber einmal zu viel als zu wenig den Browser neu starten !

JS-Dateien:

wenn JS-Dateien inkludiert werden, so diese vorher einzeln testen, ob fehlerfrei (in einem Dokument, das NUR das include enthält auf Scriptfehler testen).

erst wenn JS-Dateien syntaktisch sauber, sind die Fehlermeldungen auf das Dokument bezogen, das die JS-Dateien inkludiert. Ausnahme: Fehler einer JS-Datei z.B. innerhalb einer Funktion der ERST mit Ausführung der Funktion erkannt wird, also MIT Laufzeit des Dokumentes, das die JS-Datei inkludiert.

Die Fehleranalyse wird im Dokument mit JS-Dateien immer ohne Referenz der Fehler auf die JS-Quelldatei vollzogen und dafür immer bezüglich Dokument. Zeilennummern des Fehlers können sich aber auf JS-Dateien beziehen, ohne Hinweis, dass der Fehler in der JS-Datei liegt und schon gar nicht in welcher JS-Datei, obwohl der Parser das Inkludieren der JS-Dateien ausführt, also Informationen zur JS-Datei vorliegen könnten.) Mit anderen Worten: Der IE ist ohne Fremdttools bezüglich modularer Programmierung so gut wie gar nicht effektiv nutzbar, dafür ideal für Try and Error-Programmierung als Rätselraten im Go-To-Stil.

Endlosschleifen können zum Browserabsturz führen:

Schleifen wie for sind schnell und CPU-nah implementiert (analog zu Assembler-Implementation)



also damit Ressourcen allozierend:
 Sollte das Timing des Browsers und Windows gefährdet sein, so bricht der Browser im
 Regelfall den rechenintensiven Task ab, aber eben nicht immer.
 Endlosschleifen sind unbedingt zu vermeiden !
 Schleifen sind im Umfang zu minimieren:
 Lieber weniger verschachtelte for-schleifen, dafür andere Programmlogik.
 Die Laufzeit von Anweisungen innerhalb einer Schleife ist zu minimieren:
 Z.B. keine Rekursion innerhalb einer Schleife.

Rekursionen mit oder ohne Timer (window.setTimeout(), window.setInterval()):

Timer-Routinen belasten die Timer von Windows, das nur begrenzte Ressourcen hat.
 Z.B. kann die Uhr von Windows beeinflusst werden.
 Timer-Routinen mit Schleifen wie for sind daher besonders laufzeitempfindlich !

Lokale Variablen werden mit jedem Betreten der rekursiven Funktion neu erzeugt.
 Um Laufzeit zu verkürzen und Ressourcen zu sparen, sollte man lokale Variablen
 durch globale ersetzen. Die Rekursion nutzt dann bereits vorhandene
 und sich nicht ändernde Variablen, also Zeiger und ist damit schneller.
 Globale Variablen werden mit Parsen des Quelltextes generiert, also vor dem
 Parsen aller Funktionen.

Beispiel:

```
var a=1000;
var b=1; // Zähler
var c=0; // Timeout-ID
var d=100; // Wartezeit in Millisekunden

function x()
{
    b++;
    a-=b; // identisch mit a=a-b;

    c=window.setTimeout('x()',d); // erneuter Aufruf von x nach Wartezeit
                                   // also 'x()' wie per eval('x();') ausführen
}
```

Analog sollte bei Rekursion anstelle return ebenfalls eine globale Variable verwenden.

Zur Eineindeutigkeit von globalen Variablen einer Funktion empfiehlt es sich,
 in den Namen der Variablen den Namen der Funktion einzubauen.
 Da die Funktion eineindeutig, so sind es die Variablen auch.

Beispiel:

```
var IchTeste_Zaehler=0;
function IchTeste()
{IchTeste_Zaehler++;}
```

Zur Vereinfachung sollten Funktionen mit Beginn des Namens mit einer Abkürzung
 versehen werden, der auf Funktion hinweist. Analog dazu bei Variablen.

Beispiel:

```
var X_IchTeste_Zaehler=0;
function Y_IchTeste()
{X_IchTeste_Zaehler++;}
```

Stack Overflow:

Diese Fehlermeldung weist darauf hin, dass sich eine Funktion selbst aufruft
 OHNE window.setTimeout() oder window.SetInterval().

Da beim Aufruf verfügbare die Argumente geparkt werden und diese pro Aufruf
 referenziert werden müssen, legt die Windows-Scriptmaschine diese
 auf dem Programm-Stack ab (Speicherbereich analog beim Programm aus Compilation).

Ist der Stack voll, kommt diese Meldung.

setTimeout(x,y)

x kann Zeiger (z.B. Feldelement in dem ein Funktionszeiger liegt)
 sein, der wie folgt kodiert werden muss:

'x()'



setTimeout ruft also eval auf

Daher kann auch eine dynamisch erzeugte Funktion verwendet werden.

Returnwert bei Rekursion:

```

var X=-1;
function test()
{
    X++;
    if (X < 5){ window.setTimeout('test()',1000);}
    else{return X;}
}

var Z=test();    // es wird nicht gewartet, bis die Rekursion beendet ist, sondern sofort Z belegt
                //      Die Rekursion läuft parallel weiter und erzeugt nach 5 Sekunden den
                //      Returnwert,
                //      also NACH Z=test();
                // Da nicht gewartet wird, liefert Z einen null-Zeiger
                //      alert(test() == null); liefert true

                // Damit gilt: Z=test(); kann für Rekursionen NICHT verwendet werden
                //      Grund: Die Funktion endet korrekt und ruft sich nach 1 Sekunde wieder
                //
                //      (neue Instanz)
                //
                //      also wäre das 5x der Funktionsaufruf
                //      wobei NUR der letzte return ausführt.

                // Lösung: test () muss eine neue Funktion aktivieren, die den gesamten
                // Programmablauf hinter dem
                //      Aufruf der Rekursion beinhaltet. Die alte Funktion muss also
                //      direkt nach dem Start der
                //      Rekursion enden !

alert(test() == null);

```

auf

Timeoutwert der Rekursion:

```
window.setTimeout('test()',TimeoutWert);
```

TimeoutWert muss mindestens so hoch sein wie die Laufzeit der Funktion:
Funktion muss enden können OHNE dass die vor Ende erneut aktiviert wird.

Grund: Paralleles Aktivieren bedeutet Dateninkonsistenz für alle globale Variablen, die von der Funktion beschrieben werden zum Zweck des Lesens im nächsten Aufruf der Funktion. Dann dürfen globale Daten nicht parallel verwaltet werden, also z.B. kein Lesen im nächsten Aufruf wenn Daten noch gar nicht geschrieben sind im aktuellen Aufruf der Funktion.

Der TimeoutWert ist in Millisekunden, der von Windows unabhängig von der CPU-Geschwindigkeit verwendet wird.

Verwendung von eval():

Bei Verwendung von eval() sind Scriptfehler z.T. erst zur Laufzeit ermittelbar.
Grund: Das Parsen von Script während der Laufzeit.

```
return (eval-ausdruck mit return-Anweisung);
```

Es wird erst innere return-Anweisung von eval abgearbeitet
und dann vom äusseren return entgegengenommen

```
Bsp.: eval("return window.event.returnValue");
```

Ist (z.B. in einem Eventhandler) eine äussere return-Anweisung in der Funktion
zwar notwendig zu kodieren,
aber nicht kodiert WORDEN,
dann kann eine Fehlermeldung, dass ein return ausserhalb der Funktion liegt



Argumentenliste einer Funktion:

Die Argumentenliste einer Funktion wird ERST und mit JEDEM Ausführen der Funktion geprüft

Beispiel:

```
function x(X00,X01)
{
    X00=1;
    X01=2;
    X03=3;
}
```

verursacht Fehlermeldung, dass X03 nicht verfügbar ist

Werden Argumente nicht mit Wert belegt, so werden die Variablen der Argumente nicht erzeugt, sind also

null

Argumente sind null-Zeiger wenn kein Wert mit Funktionsaufruf übergeben wird.

Beispiel:

```
function test(x)
{alert(x==null);}
```

```
test();           // liefert true
test(1);         // liefert false
```

Es müssen also bei korrekter Programmierung die Argumente auf geprüft werden

ERST auf != null

DANN auf Wertbereich.

Achtung: Der Datentyp des Argumentes wird mit dem Wert, dem das Argument erhält, festgelegt

Das Argument ist ein Zeiger, der auf einen Speicherbereich mit einem Wert,
der Daten nach JScript-zulässigen Typen hat.

Da mit JEDEM Aufruf der Funktion die Argumentenliste geprüft wird, kostet das Laufzeit.

Gerade bei Rekursionen per Funktion sollte man daher

anstelle der Argumentenliste

globale Variablen verwenden, denn die wurden bereits geparkt und als

Zeiger hinterlegt.

Beispiel:

```
var a=1000;
var b=1; // Zähler
var c=0; // Timeout-ID
var d=100; // Wartezeit in Millisekunden

function x()
{
    b++;
    a-=b; // identisch mit a=a-b;

    c=window.setTimeout('x()',d); // erneuter Aufruf von x nach Wartezeit
                                   // also 'x()' wie per eval('x();') ausführen
}
```

Analog sollte bei Rekursion anstelle return ebenfalls eine globale Variable verwenden.

Returnwert einer Funktion:

Eine Funktion kann immer ein return kodiert haben, da der Returnwert vom

Aufrufer der Funktion ausgewertet werden kann, aber nicht muss
(Auswertung per Ausdruck mit Referenz auf Funktion z.B. Y=test();)

Wird ein returnwert geliefert UND wird dieser auch ausgewertet, so muss

der Aufrufer der Funktion, die return liefert, warten, bis der
Returnwert geliefert wurde: Dass ein Returnwert geliefert werden
soll, weiss der Aufrufer VOR Aufruf der Funktion nur deshalb, weil
die Auswertung kodiert wurde. Ob die aufgerufen Funktion wirklich
einen Returnwert liefert, steht in den Sternen, da JScript auf
Parserbasis arbeitet, also der Funktionswert nur dann erzeugt wird,
wenn return vorgefunden wird (JScript kennt keine PROCEDURE und
FUNCTION wie z.B. bei Borland Pascal, also keine Unterscheidung
zwischen Funktionen, die Funktionswert nicht liefern dürfen
bzw. immer liefern müssen).



Will man, dass der Code nach einer Funktion nicht parallel zur Funktion
abgearbeitet wird, dann gilt:
Funktion darf keine Rekursion (mit/ohne Timer) sein
Funktion muss return kodiert haben
Aufruf der Funktion mit Referenz auf Funktion und damit auf den Returnwert.

Function und Stack Overflow:

Diese Fehlermeldung weist darauf hin, dass sich eine Funktion selbst aufruft
OHNE window.setTimeout() oder window.SetInterval().

Da beim Aufruf verfügbare die Argumente geparkt werden und diese pro Aufruf
referenziert werden müssen, legt die Windows-Scriptmaschine diese
auf dem Programm-Stack ab (Speicherbereich analog beim Programm aus Compilation).

Ist der Stack voll, kommt diese Meldung.

Eine Eigenschaft eines Objektes kann per null-Zeiger-test abgefragt werden ohne Scriptfehler

Bsp.: <INPUT ID='ID_INPUT'> ohne TYPE-Attribut hat keine Eigenschaft tagName

if (ID_INPUT.style.test != null)

Achtung: Es ist kann automatisches Prototyping aktiv sein

Bsp: ID_INPUT.style.test='33'
alert(ID_INPUT.style.test); liefert 33

Wenn <SCRIPT ...>

<!--

kodiert wurde, dann sollte auch

//-->

</SCRIPT>

kodiert werden

Fehlernummer	Info
1002	allgemeiner Syntax Fehler
1003	'.' erwartet
1004	',' erwartet
1005	('' erwartet
1006)' erwartet
1007] ' erwartet
1008	{' erwartet
1009	}' erwartet 1012 '/' erwartet
1010	Bezeichner erwartet
1011	'=' erwartet
1014	falsches Zeichen
1015	Stringkonstante falsch
1016	Kommentar hier nicht zulässig
1018	'return' Anweisung liegt außerhalb einer Funktion
1019	break Anweisung außerhalb der Schleife unzulässig
1020	continue Anweisung außerhalb der Schliefe ist unzulässig
1023	Hexadezimal-Ziffer erwartet
1024	'while' erwartet
1025	ein und dasselbe Label (Sprungmarke) wurde mehrmals definiert
1026	Label (Sprungmarke) nicht gefunden
1027	'default' kann nur genau 1 mal in der switch Anweisung kodiert werden
1028	Bezeichner oder String oder Number erwartet
1029	'@end' erwartet
1030	bedingtes Parsen ist nicht aktiv
1031	Konstante erwartet
1032	'@' erwartet
1033	'catch' erwartet
1035	throw Anweisung muss hinter dem Ausdruck auf gemeinsamer Quellcode-Zeile folgen

3.10.3. Abfangen von Runtime Fehlern (Laufzeitfehlern)

siehe auch error JScript-Objekt des Internet Explorer für private Run-Time-Error

try catch finally

Fehlerbehandlung in Script (nicht Ereignisse von Objekten !)
Verschachtelung möglich



Fehler: Runtime Error
oder per throw Anweisung erzeugter Error

Syntax:

```
try           {tryStatements}
[catch(exception) {catchStatements}]
[finally      {finallyStatements}]
```

tryStatements können Fehler erzeugen, der abgefangen werden soll

exception freier Bezeichner als Platzhalter für den Fehler enthält, der aus der Abarbeitung von tryStatements resultiert
Variable wird automatisch gefüllt

catchStatements exception auslesen und daraufhin eine Reaktion ausführen
werden nur abgearbeitet, wenn Fehler auch wirklich auftrat in der Abarbeitung von tryStatements

finallyStatements wird immer abgearbeitet, egal ob ein Fehler in der Abarbeitung von tryStatements und / oder catchStatements auftrat oder nicht

Beispiel:

```
function Anzeige(Kette)
{ alert(Kette); }

try { Anzeige("try1");
{
    try { Anzeige("try 2"); }
    catch(e) { Anzeige("catch2 " + e); }
    finally { Anzeige("finally2"); }
}
catch(e) { Anzeige("catch1 " + e); }
finally { Anzeige("finally1"); }
```

throw

freie Fehlerbedingung erzeugen für try...catch...finally
throw im try kodieren als freie Fehlerbedingung
catch auswerten

Syntax:

```
throw exception;
```

exception Wert oder Variable

Beispiel:

```
function Anzeige(Kette)
{ alert(Kette); }

try { Anzeige("try1");
{
    try
    {
        throw "Das ist eine Fehlerbedingung";
        Anzeige("try 2");
    }
    catch(e)
    {
        if ( e == "Das ist eine Fehlerbedingung" )
        { Anzeige("catch2 " + e); }
    }
    finally { Anzeige("finally2"); }
}
catch(e) { Anzeige("catch1 " + e); }
finally { Anzeige("finally1"); }
```

onerror

Ereignis ausgelöst, wenn während des Ladens eines HTML-Dokumentes ein Syntaxfehler auftritt
eines Bildes ein Fehler auftritt
der Abarbeitung von Scripten ein Runtime-Error auftritt

sämtliche Fehlermeldungen unterbinden per

```
var RetteOnErrorHandler = window.onerror;
```



```
window.onerror = null;
```

Eventhandler muss wie folgt kodiert werden:

```
function freier_name_fuer_onerror_behandlung
( error_erklaerung_string,
  url_des_html_dokumentes_als_string,
  zeilen_nr_des_errors_im_html_dokument
)
{
    .....
    return true;           // nur wenn true geliefert, dann
                           // wird die Browsereigenen
                           // onerror-Behandlung
                           // unterdrückt
}
```

pro Fehler ein Aufruf des Eventhandlers
--> Folge von Fehlern, also Folge von Aufrufen

Die Script-Debugger-Aufforderung ist nur in den Eigenschaften des
Browsers abschaltbar

Beispiel 1:

```
<SCRIPT ...>
<!--
function eigenes_fehler_fenster(meldungs_text,url,zeilen_nr)
{
    // Fehlermeldung bilden
    var url_als_kette=url.toString();
    var zeilen_nr_als_kette=zeilen_nr.toString();
    var meldung_komplett=meldungs_text + url_als_kette + zeilen_nr_als_kette;

    // Meldungsfenster
    var fenster=window.open();
    with (fenster.document)
    {
        open("text/html"); // HTML-Dokument im Fenster erzeugen

        writeln("<HTML><HEAD><TITLE>Private Errormeldung</TITLE></HEAD>");
        writeln("<BODY><H1>Fehlermeldung</H1>");

        writeln(meldung_komplett);

        writeln("<FORM><INPUT TYPE=BUTTON VALUE='OK'
onclick='self.close()'>");
        // ist EINE Zeile

        // Button anklicken, damit das Meldungs-Fenster geschlossen wird
        close(); //HTML-Dokument schließen
    }
    return true;           // muss true liefern für window.onerror
}

var RetteOnError=window.onerror;

window.onerror= eigenes_fehler_fenster; // NICHT onError kodieren !
// keine () kodieren, da sonst die Funktion
// sofort ausgeführt wird !!
// window.onerror verlangt die Zuweisung
// von true für Abschaltung des
// Standard-onerror

// -->
</SCRIPT>
...
<BODY>
...
</BODY>
```

Beispiel 2:

```
<SCRIPT>
function FehlerAufspueren (MeldungString, UrlString, ZielenNummerString)
{
```



```

        ID_Div.innerHTML += "<B>Fehler erkannt</B>";
        ID_Div.innerHTML += "Error: " + MeldungString + "<BR>";
        ID_Div.innerHTML += "Line: " + ZielenNummerString + "<BR>";
        ID_Div.innerHTML += "URL: " + UrlString + "<BR>";

        return false;
    }

    window.onerror=FehlerAufspueren;    // ohne () kodieren
</SCRIPT>
<BODY>
<DIV ID="ID_Div"></DIV>
<BODY>

```

Beispiel 3 für Bild:

```

<SCRIPT>
    var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

    function Laden()
    {
        ID_Div.innerHTML=Kette;                // wird sofort geparkt, also IMG-Tag ausgeführt
                                                //      ( anstelle von eval()
                                                //      bzw. document.write() )

        ID_IMG.src="";
        ID_IMG.style.display="block";

        ID_IMG.onerror= IMGAltTextAufFehlerMeldung;    // ohne ()

    }

    function IMGAltTextAufFehlerMeldung ()
    {
        ID_IMG.alt="Das Bild konnte nicht geladen werden.";
        return true;
    }
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>

```

4. Objekte in Javascript und im Browser

Ansatz und Begriffe zur objektorientierten Programmierung eines HTML-Dokumentes mit Javascript

HTML-Dokument und seine HTML-Elemente

Das HTML-Dokument setzt sich aus HTML-Elementen (Tags) zusammen. Um diese mit dem Dokument wirksam werden lassen zu können (z.B. per Anzeige im Browserfenster), muss der Browser wissen, **ob und wie** er ein HTML-Element wirksam werden lassen kann bzw. darf oder muss, also **welche** Möglichkeit das Element zulässt.

Browser und Scriptmaschine

Der Browser muss sich dabei auf die Scriptmaschine berufen, die dem Browser genau diese Möglichkeit zum HTML-Element vorschreibt, also **vordefiniert**. Die Scriptmaschine parst das HTML-Element im HTML-Quellcode und teilt dem Browser genau diejenigen Anweisungen mit, die das HTML-Element wirksam werden lassen. Die Scriptmaschine kann HTML-Code und z.B. auch Scriptcode parsen. HTML und Javascript sind Scriptsprachen.

HTML-Dokument, HTML-Elemente und Scriptmaschine

Die Scriptmaschine hat einen bestimmten Umfang an Wirkungen für HTML-Elemente. Der Umfang zu jedem HTML-Element ist vordefiniert und damit auch die Reaktion des Browsers auf ein HTML-Element, das z.B. im Browserfenster per Anzeige wirksam werden soll. Diese gilt für alle HTML-Elemente des Dokumentes, das mit seinen Elementen sozusagen lebendig wird.

aktives HTML-Dokument und aktive HTML-Elemente

Sind HTML-Elemente wirksam, so spricht man von der **Laufzeit** dieser **Elemente** und damit von der **Laufzeit des Dokumentes**, das dann auch als **aktives Dokument** mit **aktiven Elementen** bezeichnet wird.

HTML-Element und seine Daten als Komponente des Elementes

Jedes HTML-Element hat seine Daten. Soll die Wirkung eines Elementes z.B. die **Anzeige von Text** sein, dann ist dieser Text eine einzelne Date des Elementes. Daten sind also Teil des HTML-Elementes, also dessen Komponenten.



Datenspeicherung und Zeiger

Daten werden über das Betriebssystem im Hauptspeicher (RAM) des PC verwaltet. Der Ort im RAM, also dort, wo die Daten liegen, wird über eine Speicheradresse erreichbar. Eine Adresse im Speicher kann auch als **Zeiger** auf eine Speicherstelle bezeichnet werden. Ein Zeiger ist also der Begriff für eine Datenadresse im Speicher, z.B. der Adresse des Textes eines HTML-Elementes.

HTML-Element und Zeiger auf Daten

Daten zu einem HTML-Element sind nur dann ansprechbar, wenn das HTML-Element **aktiv** ist. Es wäre daher für den Programmierer sehr praktisch, wenn das Ansprechen des **aktiven** HTML-Elementes gleichzeitig die Daten im Speicher **mit** ansprechen würde. Es gibt noch andere wichtige Gründe:

Das Betriebssystem verwaltet Daten der Instanz automatisch und ohne Hinzutun des Programmierers.

Diese Verwaltung betrifft die **physische** Ablage der Daten im Speicher und deren physische Manipulation.

Der Programmierer muss sich nicht um die physische Speicheradressierung kümmern, sondern verwaltet den Speicher **logisch**, also nur im Zusammenhang mit der Programmierung.

Die Scriptmaschine nutzt die Schnittstelle zum Betriebssystem (bzw. ist z.T. eine Komponente des Betriebssystems) und die Schnittstelle zum Programmierer, also Javascript.

HTML-Element und andere Komponenten

Bisher wurden nur die Daten zum HTML-Element, z.B. der Text, betrachtet. Aber bekanntlicherweise hat ein Text auch ein Aussehen (Layout) wie fett oder unterstrichen. Das Layout ist also eine Komponente des HTML-Elementes.

Für den Fall, dass z.B. Einfügen oder Ausschneiden von Text möglich sein sollen, muss das HTML-Element eine Komponente bekommen, die solche Aktionen realisieren.

Typische Komponenten zu einem HTML-Element sind also die Komponenten Daten (z.B. Text), Eigenschaft (z.B. Layout) und Aktion (z.B. Text ausschneiden). Diese Komponenten werden in der Objektsicht auf das HTML-Element erneut auftauchen, dort aber z.T. anders bezeichnet.

HTML-Element als Instanz (Komponentensammlung)

Wie man sieht, reicht damit ein Zeiger nur auf Daten nicht aus. Das aktive HTML-Element muss erweiterbar sein, also zur **Laufzeit** des HTML-Elementes das Layout und die Textveränderung mitgeteilt bekommen können. Ergo sind Layout und Textveränderung **ebenfalls** im Speicher abzulegen. Es muss also 3 Zeiger geben: 1 Zeiger auf Daten, 1 Zeiger auf Layout und 1 Zeiger auf die Textveränderung als Aktion des aktiven HTML-Elementes.

In HTML werden Layout und Aktion durch die Attribute **im** HTML-Element (Tag) beschrieben, wobei anstelle von HTML die scriptgesteuerte Festlegung von Layout und Aktion sehr oft umfangreicher sein kann.

Um den Programmierer die gleichzeitige Verwaltung von 3 Zeigern zu ersparen, wird **ein Zeiger** erzeugt, der auf die o.g. 3 Zeiger automatisch verweist. Dieser verweisende Zeiger wird **Instanz** genannt. Ein **aktives** HTML-Element wird daher auch als **Instanz** eines HTML-Elementes bezeichnet.

Eine Instanz ist die Schnittstelle, die eine Erzeugung und/oder Veränderung von Daten, Layout und Aktion zur **Laufzeit** zulässt. Die Instanz selbst muss daher ebenfalls im Speicher liegen.

Für den Programmierer ist also die Instanz

eine Sammlung von Komponenten des aktiven HTML-Elementes, die alle im Speicher liegen
und ein Zeiger auf diese Komponenten, also auf das aktive HTML-Element, wobei der Zeiger ebenfalls im Speicher liegt.

Anhand der Instanz kann das aktive HTML-Element zu seiner Laufzeit manipuliert werden. Instanz und aktives HTML-Element sind für den Programmierer synonym.

Erzeugung und Verwaltung einer Instanz

in Javascript:

Aufgrund der Tatsache, dass eine Instanz und ihre Komponenten nur zur Laufzeit existieren, liefert Javascript dem Programmierer die Möglichkeit, im Quellcode anstelle von Zeigern (deren Werte zum Zeitpunkt der Quellcode-Erstellung nicht bekannt ist, da die Kodierung ja außerhalb der Laufzeit des HTML-Elementes und seiner Instanz stattfindet) eine **greifbare** Größe zu nutzen, anhand derer die gesamte Manipulation der Instanz und deren Komponenten programmierbar ist. Diese Größe wird **Variable** genannt.

Eine Variable besteht aus den Komponenten

Namen (Bezeichner) der Variablen, den der Programmierer z.T. selbst festlegen kann,
und Inhalt (Wert) der Variablen, den der Programmierer z.T. selbst festlegen kann.

Wie man leicht sieht, ist eine Variable ein Komponentensammlung und tatsächlich wird sie anhand des Javascript-Objektes "var" instanziiert, was die gleichnamige Javascript-Anweisung "var" automatisch besorgt.



Der Programmierer weist im Javascript-Quellcode der Variablen einen Wert zu, der aus der Anweisung zur Instanzierung des HTML-Elementes resultiert, im Speicher liegen muss und somit einen Zeiger besitzt. Die Variable wird also **Zeigervariable** genannt. Die Kodierung der Anweisung zur Instanzierung ist somit der **Platzhalter** für den Zeiger, der erst zur Laufzeit gebildet wird.

Anhand des Variablennamens kann der Programmierer auf die Instanz zugreifen. Dass die Variable zur Laufzeit selbst im Speicher liegt, interessiert den Programmierer nicht. Und dass die Variable zur Laufzeit den Zeiger auf die Instanz beinhaltet, erfreut den Programmierer, aber interessiert ihn letztendlich auch nicht. Wichtig ist nur der **Variablenname**, der vom Programmierer natürlich sinnvoll ausgewählt sein sollte (falls überhaupt möglich). Dieser Variablenname ist also der Dreh- und Angelpunkt beim Zugriff auf die Komponenten des aktiven HTML-Elementes, wobei dem Programmierer völlig egal ist, wo was wie im Speicher liegt.

Für den Programmierer wird somit der **Name** der Zeigervariablen synonym zur **Instanz** des **aktiven** HTML-Elementes, also synonym zum aktiven HTML-Element und seinen Komponenten, die er mit der Zeigervariablen einzeln ansprechen kann.

Damit wird klar, dass Komponenten eines HTML-Elementes ebenfalls einen Namen haben müssen, der im Javascript-Quellcode kodierbar ist, also außerhalb der Laufzeit des aktiven HTML-Elementes. Diese Komponentennamen sind z.T. durch die Scriptmaschine genau definiert. Der Programmierer muss also darüber Kenntnis haben.

durch die Scriptmaschine:

Die Scriptmaschine erzeugt und verwaltet diverse interne Instanzen, auf die der Programmierer im Javascript-Quellcode nur dann zugreifen kann, wenn ihm der vordefinierte Namen der jeweiligen Instanz bekannt ist. Dieser Name fungiert dann **wie** eine bereits erzeugte und gefüllte **Zeigervariable**, ist also der Platzhalter für die jeweilige Instanz zur Laufzeit.

durch den Browser:

Der Browser erzeugt selbst keine Instanz, sondern nutzt dabei die Scriptmaschine: Diese parst im HTML-Quellcode den Tag zum HTML-Element und ermittelt, welche Attribute vorliegen. Anhand dieser und dem Tag erzeugt die Scriptmaschine die Instanz zum HTML-Element, welches damit zum aktiven HTML-Element wird.

HTML-Element im Browser

Der Browser ist die z.T. visuelle Schnittstelle zum User. Die Verarbeitung eines HTML-Elementes im Browser vollzieht **nur** die Scriptmaschine z.T. in Verbindung mit Komponenten des Betriebssystems (z.B. ActiveX unter Windows). Das Ergebnis der Verarbeitung ist z.B. die Anzeige des HTML-Elementes im Browserfenster. Visuell wird also für den User der Browser aktiv, der die Instanz, die Scriptmaschine und z.T. Komponenten des Betriebssystems benötigt. Erst dadurch ist der Browser in der Lage, das HTML-Element visuell zu verarbeiten, also dem User zugänglich zu machen.

Z.B. Abarbeitung eines HTML-Elementes mit Text:

Die Instanz besitzt intern 3 Zeiger:

- 1 Zeiger für die Textdaten im Speicher
- 1 Zeiger für das Layout des Textes
- 1 Zeiger für die Aktion zum Text also der Algorithmus des Anzeigens

Anhand der Instanz ist der Browser überhaupt erst in der Lage, dem Text des HTML-Elementes (Daten) ein Layout (Eigenschaft) zu verpassen und dann den Text anzuzeigen (Aktion).

HTML-Element und Eigenschaft

Ein Text als Datum eines HTML-Elementes kann natürlich in verschiedenen Erscheinungsformen angezeigt werden, z.B. unterstrichen.

Ein HTML-Element kann eine **Eigenschaft** z.B. "unterstrichen" besitzen.

HTML-Element und Methode

Die Anzeige von Text ist eine Aktion zum HTML-Element. Anstelle von **Aktion Anzeige** wird auch von **Methode Anzeige** gesprochen. Anstelle von **anzeigen** wird von **rendern** gesprochen. Ein HTML-Element wird durch sein **rendern** sichtbar. Nicht alle HTML-Elemente sind **renderbar**.

Ein HTML-Element kann eine **Methode** z.B. "Anzeige von Text" besitzen.

Scriptmaschine und Implementation

Die Anzeige des Textes, also die Wirksamkeit eines HTML-Elementes per Methode, wird von der Scriptmaschine beeinflusst und das in dem Umfang, wie die Scriptmaschine die Methode "Anzeige von Text" überhaupt zulässt bzw. in Art und Weise realisiert. Zulassung und Art und Weise werden auch **Implementation** der Aktion genannt. Die Scriptmaschine **kann** zur Textanzeige eine Methode besitzen, muss aber nicht. Die Methode ist also in der Scriptmaschine implementiert oder nicht. Analog gilt das auch für eine Eigenschaft, die in der Scriptmaschine implementiert sein kann, aber nicht sein muss. Über Implementationen entscheidet der Hersteller der Scriptmaschine.



Objekt

Der Begriff "Objekt" ist z.B. synonym zur Komponentensammlung eines HTML-Elementes, geht in seiner Bedeutung aber weiter: In Javascript und per Scriptmaschine sind **alle** deren Elemente grundsätzlich Objekte, also nicht **nur** die HTML-Elemente.

Objekt und Instanz

Ein Objekt kann nur als Instanz verarbeitet werden, egal ob per Scriptmaschine oder per Javascript.

interne Objekte des Browsers

Die Scriptmaschine implementiert ihre Objekte als Instanzen in den Browser, welche dann **browserinterne** Objekte genannt werden. Die Implementation findet während der Laufzeit des Browser statt.

Z.T. sind im Browser Komponenten des Betriebssystems ebenfalls als Objektinstanzen während der Browserlaufzeit zusätzlich implementiert.

Verwaltung von HTML durch Instanzen

Das HTML-Dokument **und** dessen HTML-Elemente werden durch die Scriptmaschine komplett zu Instanzen von Objekten umgewandelt.

Objekt und Vererbung

Wie im Layout eines HTML-Dokumentes ersichtlich, können HTML-Elemente andere Elemente einschließen. Das einschließende Element wird dabei **Container** genannt.

Manchmal ist es erwünscht, dass ein im Container liegendes Element sich **ähnlich** oder zeitlich synchron oder angepasst zum Container verhält. Der Container muss dann also Einfluss auf sein inneres Element haben können, wobei dem inneren Element der "eigene Wille" erhalten bleibt, welcher aber den des Containers zu beachten hat (Der Internet Explorer unterstützt vielfältige Synchronisierung und Filter).

Wenn der Container das innere Element beeinflussen soll, so muss er dem Element mitteilen, wie es sich zu verhalten hat. Das Verhalten ist also eine Methode des Containers. Sogar eine Eigenschaft zum Zweck der Anpassung des inneren Elementes kann der Container übergeben. Der Container vererbt also seine Methode und/oder Eigenschaft an das innere Element. Das innere Element wird durch Erbschaft um eine Methode und/oder Eigenschaft des Containers erweitert und kann diese ebenfalls erweitern, falls es die Scriptmaschine zulässt. Vererbung bedeutet damit die Erweiterung des **Objektes** des Elementes im Container. Bei der Vererbung ist immer maßgebend, was der Container **vorgibt**, ohne dabei dem inneren Element sein Eigenleben zu nehmen. Diese Vererbung entspricht einem **Eltern-Kind-Verhalten**. Der Container ist also **Eltern** zum inneren Element, seinem **Kind**. Vererbung wird **mit** dem Objekt instanziiert.

Objekthierarchie und Baumstruktur

Aufgrund der Vielfalt im Layout des HTML-Dokumentes ist es nötig, dass Objekte mehrere Eigenschaften, Methoden und sogar Kinder haben können. Die so entstehende **Objekthierarchie** hat also eine **Baumstruktur**. Z.B. besitzt das HTML-Dokument eine Baumstruktur, da es als Container aller HTML-Elemente dient und durch diese lebendig wird. Das HTML-Dokument als Container ist die **Wurzel** der Baumstruktur.

Die Wurzel der Baumstruktur nennt man auch **Root**.

Objekthierarchie und Punktnotation in Javascript

Um in dieser Baumstruktur die Übersicht zu behalten, wird in Javascript die Punktnotation benutzt. Jede Hierarchie-Ebene wird also in der Scriptkodierung von Objekten durch einen vorausgehenden Punkt markiert. Keinen Punkt erhält die unterste Ebene, also die Wurzel (Root).

Scriptmaschine und Objekthierarchie

Die Möglichkeit einer Vererbung und die Hierarchie der Objekte sind in der Scriptmaschine **implementiert**.

Die Scriptmaschine bestimmt damit

welche Objekte existieren (Art der Objekte, auch **Objektklasse** oder **Objektyp** genannt),

welche Eigenschaften und Methoden zum jeweiligen Objekt existieren,

welche Objekte vererben bzw. erben

welche Objekte vererbte Methoden bzw. Eigenschaften erweitern dürfen

und welche geerbte Methoden bzw. Eigenschaften verändert werden dürfen.

Daran muss sich der Programmierer halten, der damit nicht nur Kenntnisse zu Javascript sondern auch zur Scriptmaschine und den Objekten je nach Version der Scriptmaschine haben muss, in der Hoffnung, dass die Hersteller der Scriptmaschine die Abwärtskompatibilität der Versionen aufrecht erhalten (auch bezüglich der Browser und deren möglichen Scriptmaschinen-Versionen sowie bezüglich der diversen HTML- und CSS-Versionen und deren Umsetzung in den jeweiligen Browser- und Scriptmaschinen-Versionen).

Browserfenster als Objekt

Im Zuge der Vererbung muss das Browserfenster ebenfalls ein Objekt sein. Das Browserfenster ist der virtuelle Container des HTML-Dokumentes, das in ein Fenster geladen wird. Browserfenster und HTML-Dokument sind aber komplett getrennte Objekte. Ein HTML-



Dokument muss zwar ein Fenster besitzen, es aber nicht für Anzeige etc. nutzen. Nicht jedes HTML-Dokument wird gerendert, wenn es der Programmierer so wünscht. Browserfenster und Dokument können bezüglich ihres "Willens" getrennte Wege.

Objekt und Ereignisse

Auch für die Synchronisierung von Elementen werden Ereignisse genutzt. Ein Ereignis signalisiert einen bestimmten Zustand eines Elementes, wo bei der Zustand mit seinem Auftreten einem anderen Element mitgeteilt werden kann. Im Gegensatz zur Vererbung ist aber der Ereignisfluss in beiden Richtungen möglich: Von innen nach außen **und** von außen nach innen, **oder** nur eines von beiden. Das Weiterreichen von Ereignissen **kann** durch den Programmierer ermöglicht aber auch unterdrückt werden, dagegen die Entstehung von Ereignissen z.T. nicht. Der Browser verwaltet permanent Ereignisse, von denen der Programmierer nur z.T. Kenntnis bekommt. Die Ereignisverwaltung ist z.T. in der Scriptmaschine implementiert.

Datenstrukturen und Script-Objekte

Die Kombination von Daten ist in Form eines Datenstruktur-Objektes möglich. Die Struktur ist dabei das Abbild der Menge von einzelnen Daten. Basis-Datenstrukturen werden von der Scriptmaschine durch vordefinierte Script-Objekte bestimmt. Basis-Datenstrukturen sind eine Kombination von Basis-Datentypen, die allesamt in der Scriptmaschine implementiert sind und z.T. Objektcharakter haben. Basis-Datentypen und -Datenstrukturen hängen von der Version der Scriptmaschine ab.

Anhand der Basis-Datentypen und -Strukturen kann der Programmierer weitere Datenstruktur-Objekte kreieren. Der Programmierer kann jedoch keine Basis-Datentypen und -Strukturen erzeugen.

Scriptmaschine und Javascript als Schnittstelle zum Programmierer

Javascript ist die direkte Programmierschnittstelle zur Scriptmaschine und damit zum Browser.

Die Softwareschnittstelle zum Browser ist die Scriptmaschine.

Die Softwareschnittstelle zum User ist der Browser.

Falls der User programmieren möchte, kann er HTML oder Javascript benutzen:

HTML ist die indirekte Schnittstelle zur Scriptmaschine und die visuell direkte Schnittstelle zum Browser.

Der Browser benötigt immer die Scriptmaschine, welche z.T. Komponenten des Betriebssystems benutzt.

Für den Programmierer als praktisch erweist sich die Kombination von HTML und Javascript (analog von HTML und PHP bei Datenbankanschluss).

Javascript und die Scriptmaschine sind **komplett** objektorientiert. Es wird also **grundsätzlich** mit Objekten und deren Instanzen gearbeitet. Inwieweit der Programmierer zugreifen darf, entscheidet der Hersteller der Scriptmaschine. Man spricht dann von einem herstellerspezifischen **Dialekt** der Programmiersprache Javascript.

Hinweis: Die HTML-Programmierung ist nur z.T. objektorientiert: Über das ID-Attribut bzw. NAME-Attribut, dessen Werte je einem Zeiger entsprechen, sind HTML-Elemente im HTML-Code ansprechbar. Die Benutzung dieser Attribute ist aber **wahlweise**. Es geht also z.T. auch **ohne** Zeiger.

Für Programmierer unter Windows XP mit dem Internet Explorer bitte **unbedingt** beachten: Die Scriptmaschine unter Windows XP ist wesentlich **weniger fehlertolerant** als die Scriptmaschine unter Windows 98 bei identischer Browserversion und identischem Patch-Stand der Browsersoftware. Unter Windows XP ist der Internet Explorer 6.x implementiert. JScript-Code, der unter Windows 98 einwandfrei funktioniert, muss es unter Windows XP **nicht** ! Javascript-Code, der unter Windows XP funktioniert, wird es auch unter Windows 98 tun. Mit anderen Worten: Die Scriptmaschine unter Windows XP ist bezüglich Fehler **nicht** abwärtskompatibel ! Deswegen bitte **unbedingt** den Scriptcode unter Windows XP testen !

Kompatibilität der Javascript-Dialekte

Für die Kompatibilität der Dialekte **wäre** es wichtig, dass Hersteller sich an gemeinsame Standards halten. Wie man weiß, sind Browserhersteller Konkurrenten.

Objektdesign beim Netscape und Internet Explorer

Im Falle von Netscape 6.x ist aber inzwischen eine Änderung gegenüber dessen Vorgänger 4.7x eingetreten. Es wurde mit Fähigkeiten des verbreiteten Netscape 4.7x konsequent gebrochen, was so mancher Internet-User nicht weiß, der zu dem eventuell noch hartnäckig auf "Netscape schwört" (Letzteres ist ein kontraproduktives Verhalten, dass im Kampf der Browserhersteller zum Zweck der Erzielung monopolistischer Internet-Marktanteile und Profite dem User regelrecht suggeriert wurde, wobei Microsoft langwieriger aber cleverer vorgeht: Marktanteile auf Basis der Browser-Integration in das Betriebssystem Windows und dessen Anwendungen (auch im Internet) **und** das konsequente aber z.Z. nicht standardisierte Design im Internet Explorer, das für die Integration in das objektorientierte Betriebssystem Windows wichtig ist, aber auch dem JavaScript-Programmierer beim Web-Design zum Vorteil gereicht, solange Windows benutzt wird (Linux wird hier nicht betrachtet). Nicht zu verachten ist auch der Hardware-Ressourcen-Bedarf: Während der IE mit Hochfahren von Windows z.T. instanziiert wird, muss Netscape eigene Ressourcen allokalieren, was nachteilig ist, wenn IE und Netscape auf dem Rechner aktiviert werden.

Der Netscape 6.x ist dem Internet Explorer näher als nie zuvor. Netscape realisiert damit z.T. endlich Möglichkeiten des Web-Design per JavaScript, die auch der Internet Explorer schon länger bedient.



Knackpunkt ist also die Browserprüfung auf unterstützte browsereigene Objekte. Es könnte möglich sein, dass der Netscape auch ein `document.all` abbilden kann. Netscape 6.x hat definitiv keine `<LAYER>`- und `<ILAYER>` Unterstützung mehr, vermutlich auch keine Objekt-Unterstützung von `Layer`. Die Verwendung des `<DIV>`-Tag wurde auch geändert, obwohl dieser Tag gerade für DHTML wichtig ist.

Aufgrund der unzähligen Browserversionen und -änderungen ist es nötig, die Objektfähigkeit des Browsers zu prüfen, wenn ein Objekt benutzt werden soll. Natürlich muss der Benutzer im Browser JavaScript eingeschaltet haben. Es wird tendenziell sichtbar, dass der User ohne JavaScript-Zuschaltung mit dem rein HTML-basierenden Seiten ins Abseits gerät. Der Preis für JavaScript: Es kann missbräuchlich auf Basis der Unwissenheit des Internet-Users (der eigentlich "nur surfen will") benutzt werden. Der User muss sich entscheiden, für ihm vertraute Seiten JavaScript zuzuschalten, Software wie ein Virenschanner und eine Firewall zu nutzen, oder auf Design zu verzichten.

Scriptmaschine und implementierte Objekte

Um ein Objekt nutzen zu können, ist es wichtig zu wissen, ob der Browser das Objekt überhaupt kennt, also ob die aktuelle Scriptmaschine das Objekt implementiert, also vordefiniert hat. Anstelle von vordefinierten Objekten spricht man auch von browserinternen Objekten.

Beispiel:

```
if (window.createPopup())
{ ..... } // window.createPopup() kennt nur der IE ab 5.5
```

Die Browser-Version ist immer an die Version der Scriptmaschine gebunden. Implementationen in einer jüngeren Version einer Scriptmaschine müssen vom Browser älteren Datums nicht unbedingt nutzbar sein. Dafür ist die Scriptmaschine in der Regel zu älteren Browsern abwärtskompatibel. Browserinterne Objekte hängen also von der Scriptmaschine ab, die diese Objekte implementiert. Was die Scriptmaschine nicht kennt, kann der Browser erst recht nicht wissen.

Da sich die Scriptmaschinen zum Browser unterscheiden, muss also **vor** der Nutzung eines browsereigenen Objektes bekannt sein
welcher Browserhersteller, also Name des Browsers
welche Browserversion
welche Version der Scriptmaschine.

Tipp: Das Prüfen auf ein Objekt kann mit der Zuweisung des Objektzeigers auf eine Variable verbunden werden (analog zur Kodierung des ID-Attributes im HTML-Tag: Der Wert des ID-Attributes ist ein String, der vom Browser als Zeiger auf das Objekt laut HTML-Tag verwendet wird). Die Nutzung einer Zeigervariable erhöht die Performance des Browsers.

Beispiel für den Internet Explorer:

```
//      Prüfung auf Browsertyp
//      Dieser Quellcode muss VOR allen anderen Routinen kodiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

var IE_ZeigerAufFeldAllerElementeImHTMLDokument; // Variable hier deklarieren,
//      damit sie im gesamten Scriptcode verfügbar ist
//      Variable ist damit global zum gesamten Scriptcode

if (ns)
{ // Code für Netscape }
else
{
    if (ie)
    {
        // Zeiger auf alle Elemente (Objekte) im HTML-Dokument holen
        //      Hinweis: document.all ist eine Collection, also ein Feld der Zeiger aller Elemente im Dokument
        IE_ZeigerAufFeldAllerElementeImHTMLDokument = document.all;
        // Vorteil der Variable IE_ZeigerAufFeldAllerElementeImHTMLDokument:
        // es muss später nicht erneut document.all kodiert und damit auch nicht mehr
        //      geparkt werden

        // Feld instanzieren, dass die Zeiger auf alle HTML-Elemente mit P-Tag im Dokument aufnimmt
        //      Hinweis: document.all also IE_ZeigerAufFeldAllerElementeImHTMLDokument ist ja
        //      der Zeiger auf das Feld (Collection) aller Elemente im Dokument,
        //      also die Feld-Methode .tags() benutzen, die einen Zeiger auf ein Zeigerfeld
        //      (Collection) aller P-Tags im Dokument liefert
        //      Zeigerfeld der P-Tags ist eine Teilmenge aller Elemente im Dokument
        var ZeigerAufFeld = IE_ZeigerAufFeldAllerElementeImHTMLDokument.tags("P");
        // Vorteil der Variable ZeigerAufFeld
        // es muss später nicht mehr erneut
        //      IE_ZeigerAufFeldAllerElementeImHTMLDokument.tags("P")
        //      kodiert und damit auch nicht mehr geparkt werden

        // performance-schädigend wäre auch folgende Kodierung:
        //      var ZeigerAufFeld=document.all.tags("P");
```




```

// denn dann würde das Parsen den Zeiger document.all nochmals ermitteln

// prüfen ob überhaupt ein P-Tag im Dokument ist
// Hinweis: ZeigerAufFeld ist der Zeiger auf eine Teilmenge aller Elemente im Dokument
// und zwar auf die Teilmenge aller P-Tags im Dokument
// Die Teilmenge kann nur existieren, wenn überhaupt P-Tags
// im Dokument enthalten sind.
// Der Zeiger auf die Teilmenge kann nur korrekt sein, wenn die Teilmenge
// überhaupt existiert.
// Ein Zeiger, der nirgendwo hinzeigt, hat den Wert null (nicht numerisch 0 )
if (ZeigerAufFeld!=null)
{
    // es existiert mindestens 1 P-Tag

    // Anzahl der P-Tags im Dokument ermitteln per Feld-Eigenschaft .length

    var AnzahlDerPTags = ZeigerAufFeld.length; // Anzahl ab 1

    // man könnte jetzt noch mal prüfen, ob die Anzahl > 0 ist,
    // aber es muss ja bereits mindestens 1 P-Tag-Element existieren

    // alle P-Tag-Elemente (Objekte) im Dokument im Textbereich unterstreichen
    // Hinweis: Da die Teilmenge aller P-Tags ein Feld (Collection) ist,
    // kann also mit dem Feldindex gearbeitet werden, wobei der
    // Wert des Indexes die Position des Feldelementes im Feld angibt
    // Das Abklappern aller möglichen Werte im Index erfolgt am besten per Schleife.
    for ( var Index=0; Index < AnzahlDerPTags; Index++)
        // Index immer ab 0 !!
        // Ablauf: Index nehmen und prüfen ob < Anzahl der P-Tags
        // wenn ja dann Index verwenden im Schleifenkörper
        // und nach der Verwendung
        // den Index automatisch um 1 erhöhen
        // (nach entspricht Index++
        // davor entspricht ++Index)

    {

        // Schleifenkörper: Hier den Index verwenden
        // Hinweis: . style ist das style Objekt, mit dem ein Layout
        // geändert werden kann
        // .style ist dabei wieder ein Zeiger auf das Layout des P-Tag
        // ein P-Tag unterstützt im style Objekt die Eigenschaft
        // textDecoration
        ZeigerAufFeld[Index].style.textDecoration="underline";

    }

}

}

```

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizensierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-



Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen. Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899
Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.
User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.
Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:
Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das
Patchproblem auftritt (User muss sich Telefonnummer besorgen)
Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,
z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer
von Microsoft besorgen muss bzw. zu besorgen hat, wird der User
IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.
(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popublocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popublocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,
bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popublocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende
Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popublocker hat ein Popufenster geblockt. Sie können den Popublocker deaktivieren
oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popublocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popublocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster
einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popublocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer
anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popublocker verwaltet wird.

Der Popublocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen
Webseiten (die nicht das Popup erzeugt haben).

Der Popublocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popublockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin



```

onfocusout
und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);

```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.



Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.



Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Umeine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wirdferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, undmuss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.



Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popuptmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>
(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

4.1. in Javascript vordefinierte Operationen mit Objekten

Operationen mit Objekten erfolgen in Script neben den bekannten Operatoren und Anweisungen auch per vordefinierte Eigenschaften und Methoden, die objekt-unabhängig (objekt-übergreifend) sind, aber eventuell eine objektspezifische und/oder browserspezifische Erweiterung bzw. Abänderung besitzen. Dazu kommt, dass Eigenschaften und Methoden inzwischen **browserspezifisch** als deprecated (unerwünscht, veraltet) gelten können und nur noch auf Basis von herstellerbezogener Toleranz in der Scriptmaschine implementiert sind. Von diesen (z.T. sehr oft benutzten) Eigenschaften und Methoden sollte Abstand genommen werden, oder es muss pro Browserversion geprüft werden, ob die Implementierung noch vorliegt.

4.1.1. Operationen mit Objektinstanzen (Auswahl)

4.1.1.1. Ermittlung der Objektklasse einer Objektinstanz als Zeichenkette (typeof)

Bsp: if (typeof (objekt_instanz) == "String")

Typen: "undefined"




```
"function"
"object"
"number"
"boolean"
"string"
```

Bsp: typeof 42 ergibt "number"

4.1.1.2. Vergleich von Objektinstanzen (valueOf) (Zeigervergleich)

Vergleiche erfolgen immer als Zahl im 32-Bit-Format.

Bsp: if (objekt_instanz_1.valueOf() == objekt_instanz_2.valueOf())// Zeigervergleich

4.1.1.3. Löschen einer Objektinstanz (incl. Speicherfreigabe) per null-Zuweisung

Die Zuweisung des Wertes **null**, also eines Null-Zeigers, bewirkt die Löschung der Variablen (inklusive Speicherplatz)

```
Bsp:   var variable=0;    // numerische Variable instanzieren
       variable=null;     // Zeiger löschen, also Instanz aufheben
```

Es kann auch die Anweisung delete verwendet werden, um das gesamte Objekt oder Teile davon zu löschen (inklusive Speicherplatz)

```
Bsp.:   var Wert = delete objekt_instanz.eigenschaft;

                Wert    true,    so Löschung erfolgreich
                false,   so Löschung nicht erfolgt
```

Script-Objekte sind nicht löschar.

Nur per Prototyping zum Script-Objekt hinzugefügte Eigenschaften und Methoden sind löschar.

4.1.2. Standardmethoden aller Objekte in Javascript (Auswahl)

siehe auch Script-Objekt Object

4.1.2.1. Boolean()

konvertiert einen Wert nach Boolean

Syntax:

```
[ var Zeiger = ] Boolean(Wert)
```

```
Wert    true oder false oder beliebiger Wert
        wenn 0, -0, +0, null, false, NaN, undefined oder Leerkette ""
        so wird false verwendet
        sonst wird true verwendet
        z.B. "false" wird als true verwendet
```

4.1.2.2. decodeURI() (IE ab 5.5, NS 6.x)

dekodiert einen Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde ersetzt die Methode unescape() welche deprecated ist

Syntax:

```
[ var Zeiger = ] decodeURI(Kette)
```

```
Kette    encoded Uniform Resource Identifier, der mit der Methode encodeURI () erzeugt wurde
```

```
Zeiger    auf dekodierten String
```

Beispiel:

```
alert(decodeURI("My%20phone%20 #%20is%20123-456-7890"));
// erzeugt "My phone # is 123-456-7890"
```

4.1.2.3. decodeURIComponent() (IE ab 5.5, NS 6.x)

dekodiert eine Komponente einer Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde

Syntax:

```
[ var Zeiger =] decodeURIComponent(Kette)
```

```
Kette    encoded Uniform Ressource Identifier, der mit der Methode decodeURIComponent() erzeugt wurde
```

```
Zeiger    auf dekodierten String
```

4.1.2.4. encodeURI() (IE ab 5.5, NS 6.x)

kodiert einen String als kompletten Uniform Ressource Identifier (URI):

Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx

es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen

```
., / ? : @ & = + $ - _ . ! ~ * ' ( ) #
```



Syntax:

`[var Zeiger =] eval(Kette)`

Kette String mit JavaScript-Anweisungen

Zeiger auf Ausdrucksergebnis
 nur erzeugt, wenn JavaScript-Anweisungen einen Ausdruck als letzten Teil oder nur einen Ausdruck in der Kette enthalten
 Es ist zu empfehlen, anstelle des Zeigers die Wertzuweisung auf eine Variable direkt innerhalb der JavaScript-Anweisungen zu realisieren.

Beispiele:

```
var Kette = eval("new String('2+2')"); // liefert Zeiger auf den String '2+2'
eval("var Kette = new String('2+2')"); // liefert nichts
// Kette hat den Wert '2+2'
eval("var Kette = new String('2+2'); alert(Kette);"); // liefert nichts, da alert nichts liefert
// Kette hat den Wert '2+2'

var StringLiteral = "2 + 2";
eval(StringLiteral) // liefert Zeiger auf numerische Variable
// mit Wert 4
var Wert = eval("2+2"); // liefert Zeiger auf numerische Variable
// mit Wert 4
var StringObjekt = new String("2 + 2");
eval(StringObjekt) // liefert Zeiger auf Kette "2 + 2"
```

Beispiel:

```
var w = 2;
var x = 39;
var y = "42";
var z = "42"
eval("w + x + 1"); // liefert Zeiger auf numerische Variable
// mit Wert 42
eval(y); // liefert Zeiger auf numerische Variable
// mit Wert 42
// liefert nicht Zeiger auf y, da der String
// "42" ausgewertet wird
eval(z); // liefert Zeiger auf String-Variable
// mit Wert '42'
// liefert nicht Zeiger auf z, da der String
// "42" ausgewertet wird
```

Beispiel:

```
var Kette = "var x=5;"
+ "if (x == 5)"
+ "{"
+ "alert('Meldung aus eval\nx ist 5');" // \n ist Zeilenumbruch-Zeichen
+ "x = 42;"
+ "}"
+ "else"
+ "{"
+ "x = 0;"
+ "}";

var Kette1 = "alert('Meldung aus document.write\nx ist ' + eval(Kette));"
document.write(Kette1);
```

Beispiel für Ersatz von eval() durch die Eigenschaft .innerHTML (nur IE):

```
<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette; // wird sofort geparkt, also IMG-Tag ausgeführt
                          // ( anstelle von eval()
                          // bzw. document.write() )

    ID_IMG.src="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}
```



```
function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>
```

4.1.2.8. **isFinite()**

ermittelt, ob numerischer Wert einer Instanz endlich ist

Syntax:

```
[ var Wert = ] isFinite(Zeiger);
```

Zeiger auf Instanz vom Number Script-Objekt oder Ausdrucksergebnis
auch Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY
NaN

Wert true, so Wert der Instanz ist endlich
false, so Wert der Instanz ist unendlich
immer bei
Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY
NaN

4.1.2.9. **isNaN()**

ermittelt, ob Wert einer Instanz **nicht numerisch** is (Not a Number)

Syntax:

```
[ var Wert = ] isNaN(Zeiger);
```

Zeiger auf Instanz vom Number Script-Objekt oder Ausdrucksergebnis
auch Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY
NaN

Wert true, so Wert der Instanz ist nicht numerisch
immer bei NaN
false, so Wert der Instanz ist numerisch
immer bei
Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY

Beispiele:

```
var Kette ="10.23";
var Wert =10.23;

var Ergebnis1=parseFloat(Kette);
var Ergebnis2=floatValue=parseFloat(Wert);

alert(isNaN(Ergebnis1);
alert(isNaN(Ergebnis2);
```

Hinweis: Methoden parseFloat und parseInt liefern NaN, wenn Parameter nicht numerisch ist

Bsp: var zahl=parseFloat("text"); isNaN() liefert true

4.1.2.10. **Number()**

konvertiert eine Instanz zu einem numerischen Wert (Number Script-Objekt-Wert)

Syntax:

```
[ var Wert = ] Number(Zeiger)
```

Zeiger 1 auf zu konvertierende Instanz
auch Date Objekt

Wert wenn Zeiger 1 ein Date Objekt ist, so Wert ist die Anzahl der Millisekunden seit dem
1.1.1970 0 Uhr Weltzeit (UTC), wobei Weltzeit genau GMT entspricht
wenn Datum vor dem 1.1.1970 0 Uhr Weltzeit so Wert negativ
NaN wenn Konvertierung nicht möglich ist

Beispiel:

```
var DatumJetzt = new Date();
alert (Number(DatumJetzt));
```



4.1.2.11. *parseFloat()*

String oder Literal parsen und nach Floating-point umwandeln
können enthalten

Vorzeichen + und -
Ziffern 0 bis 9
Dezimalkomma als Punkt
e oder E
Blanks (werden ignoriert)

falls andere Zeichen enthalten sind, gilt:
String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann konvertiert

Erfolg der Konvertierung ist per Methode isNaN() zu prüfen

Syntax:

```
[ var Wert = ] parseFloat(Kette)
```

Kette String oder Literal

Wert Floating-point
NaN wenn String bzw. Literal bereits mit erstem Zeichen fehlerhaft

Beispiele: gültiges Literal

```
parseFloat("3.14")
parseFloat("314e-2")
parseFloat("0.0314E+2")
```

gültiger String

```
var x = "3.14"
parseFloat(x)
```

ungültiges Literal

```
parseFloat("FF2")
```

```
alert(parseFloat("17.50 DM"));
```

4.1.2.12. *parseInt()*

String oder Literal parsen und nach Integer umwandeln
können enthalten

Vorzeichen + und -
Ziffern 0 bis 9, aber keine Vornull(en)
Blanks (werden ignoriert)
eventuelle Buchstaben A bis F

falls andere Zeichen enthalten sind, gilt:
String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann konvertiert

Erfolg der Konvertierung ist per Methode isNaN() zu prüfen

Syntax:

```
[ var Wert2 = ] parseInt(Kette[, Wert1])
```

Kette String oder Literal
muss passend zu Wert1 kodiert sein
Hinweis: oktale Escape-Sequenzen sind in ab Javascript 1.5 im Netscape 6.x deprecated

Wert1 Integer
Zahlenbasis
10 für dezimal
8 für oktal
16 für hexadezimal (Wert1 kann Buchstaben A bis F enthalten)
Standard: 0

Wert2 Integer
NaN wenn String bzw. Literal bereits mit erstem Zeichen fehlerhaft
wenn Wert1 0 ist, so
wenn Kette beginnt mit

"0x", so hexadezimale Konvertierung
"0" und nicht "0x", so oktale Konvertierung
nicht mit "0" oder "0x", so dezimale Konvertierung

Beispiel:

gültige Literale:

```
parseInt("F", 16)
parseInt("17", 8)
parseInt("15", 10)
```



```

parseInt(15.99, 10)
parseInt("FXX123", 16)
parseInt("1111", 2)
parseInt("15*3", 10)
parseInt("17")
parseInt("0x7", 16)
parseInt("0x7")
parseInt("0")
parseInt("0x11", 16)
parseInt("0x11", 0)
parseInt("0x11")

```

ungültige Literale:

```

parseInt("F")
parseInt("Hello", 8)
parseInt("0x7", 10)
parseInt("FFF", 10)
parseInt("002")

```

4.1.2.13. **String()**

konvertiert eine Instanz zu String
ist identisch mit Methode .toString()
Syntax:

```
[ var Kette = ] String(Zeiger)
```

Zeiger auch auf Date Objekt

Kette wenn Zeiger auf Date Objekt, so Datum als String geliefert (je nach Ländereinstellung des Windows)
Beispiel: Thu Aug 18 04:37:43 GMT-0700 (Pacific Daylight Time) 1983

4.1.2.14. **toString()**

Wert eines Objektes in einen String umwandeln
Methode ist nicht für jedes Objekt implementiert (aber für die meisten)
Syntax:

```
[ var Kette = ] object.toString([Wert])
```

Wert nur kodierbar bei numerischen Wert (Objekt ist Number Script-Objekt)

Basis des Wertes

z.B. 2 für dual (binär)
10 für dezimal
16 für hexadezimal

object Referenz auf den Wert eines Objektes
Javascript 1.1 liefert Objektname als Literal
Javascript 1.2 liefert Objektnotation als Literal
wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma getrennt geliefert
wenn Objekt ein Script-Objekt Boolean ist, dann "true" bzw "false" geliefert (Kleinschreibung !)
wenn Objekt ein Script-Objekt Error ist, dann die Fehlermeldung geliefert (Objekt-Wert ist der Fehlercode)
wenn Objekt ein Script-Objekt Function ist, dann Quellcode der Funktion geliefert (inklusive Funktionskopf)
wenn Objekt ein Script-Objekt Object ist, so wird geliefert: "[**object** objectname]"
mit objectname als konkreter Bezeichner der Objektklasse bzw. des Objektes
fettgedrucktes wird so geliefert wie angegeben
wenn der Bezug vor .toString() ein Wert laut ID-Attribut oder NAME-Attribut ist, dann wird die Objektklasse geliefert

Kette String

.toString() liefert 'NaN' wenn der Wert nicht nach String konvertierbar

Beispiel 1:

```

var Wert = 33,33;
alert(Wert.toString(2) + ' ' + Wert.toString(16) + ' ' + Wert.toString(10));

```

Beispiel 2:

```

<BUTTON ID="ID_Button" .... > .... </BUTTON>
ID_Button.toString() liefert "button"

```

Beispiel 3:

```

zahl=new Number("8376");
zahl.Number.toString(10) liefert "8376" mit 10 als Basis der Zahlendarstellung

```



4.1.2.15. unescape()

ab Javascript 1.5 (Netscape 6.x) deprecated und zu ersetzen durch die Methoden

decodeURI()
decodeURIComponent()

dekodiert einen mit der Methode escape() kodierten String (siehe escape())

Syntax:

```
[ var Zeiger = ] unescape(Kette)
```

Kette String aus escape()

Zeiger auf de-kodierten String

4.1.2.16. valueOf()

Konvertierung einer Objektinstanz in eine 32-Bit-Zahl also Zeigerwert z.B. zum Zweck des Vergleiches zweier Objektinstanzen per Zeigervergleich

Bsp: if (objekt_instanz_1.valueOf() == objekt_instanz_2.valueOf())

4.1.3. Standard-Eigenschaften und -Methoden aller Objekte in Microsoft JScript

Nachfolgend werden die gemeinsamen Eigenschaften und Methoden aller Objekte in JScript und der browserinternen Objekte beschrieben, also z.B. von

- Objekt arguments
- Objekt Array
- Objekt Boolean
- Objekt Date
- Objekt Enumerator
- Objekt Error
- Objekt Function
- Objekt Math
- Objekt Number
- Objekt Object (nicht Objekt object des Internet Explorer für das HTML-Tag OBJECT)
- Objekt RegExp
- Objekt String
- Objekt var

wobei es Ausnahmen gibt: Z.B. unterstützt das Objekt Math nicht die Eigenschaft .constructor, da der Scripthersteller eine private Instanz von Math verhindern will.

JScript-Objekte sind alle in JScript implementierten Objekte, also **nicht** die browserinternen Objekte des Internet Explorer.

Das Objekt, dem diese objekt-übergreifenden Eigenschaften und Methoden gehört, ist das Script-Objekt Object, welches sozusagen den Prototyp aller anderen Objekte darstellt. Das Script-Objekt Object hat im Gegensatz zu anderen Script-Objekten wie Array oder Date keinen speziellen Datentyp implementiert. Die new Anweisung liefert also einen untypisierten Zeiger. Das Script-Objekt Object ist also für den Programmierer ein **symbolisches** Objekt, das als Objektklasse (Konstruktor) per new Anweisung für die Erzeugung eines einfachen Objektes benutzt werden kann, um es per Prototyping mit gewünschten Eigenschaften und Methoden zu erweitern und damit Datentypen beliebiger Art zu implementieren. Das Objekt ist also ideal für die Erzeugung eigener und freier Datenstrukturen anhand der Basis-Datentypen und Basis-Datenstrukturen.

Die gemeinsamen Eigenschaften und Methoden tauchen in der jeweiligen Objektbeschreibung **nicht** mehr auf (außer Script-Objekt Object).

Eine Instanz eines JScript-Objektes kann per Anweisung new erzeugt werden (siehe dort):

Achtung: Der Browser kann nur Objekte verarbeiten, die er kennt, z.B. ein Array Objekt, dessen Methoden und Eigenschaften dem

Browser bekannt sind. Bei einem privaten Objekt müssen alle Eigenschaften und Methoden auf Script-Komponenten bestehen.

Jedes Objekt kann per Prototyping um Eigenschaften und Methoden erweitert werden, wenn es die Eigenschaft .prototype besitzt.

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype .

Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft .prototype .

Punktnotation zum Zeiger:

zeiger.prototype.eigenschaft

zeiger.prototype.methode

Erweiterung eines Script-Objektes per

bezeichner_script_objekt.prototype.eigenschaft

bezeichner_script_objekt.prototype.methode

Eigenschaften und Methoden müssen auf JScript-Elemente **basieren**, denn nur letztere kennt die Scriptmaschine des Browsers.



Eine Referenz auf eine Objekt-Instanz ist auch per Anweisungen `this` und `with` möglich (siehe dort).

Prinzipiell dürften andere Scripthersteller wegen der Skript-Kompatibilität die gleichen Script-Objekte und deren Eigenschaften und Methoden wie in JScript unterstützen.

Standard-Eigenschaften aller Objekte in JScript:

<code>.constructor</code>	Bezeichner einer JScript-Objektklasse (Objekttyp) oder eines privaten Konstruktors Anwendung: Ermittlung der Objektklasse/Konstruktors eines abgeleiteten Objektes Ableitung aus der Objektklasse per Anweisung <code>new</code> mit der Objektklasse als Konstruktor benötigt (siehe dort) nicht bei Script-Objekt <code>Math</code> möglich Ableitung aus privatem Konstruktor per Anweisung <code>new</code> , die den privaten Konstruktor verwendet JScript-Objektklassen sind z.B. Array Boolean Date Function
---------------------------	---

Beispiel 1 für Ableitung aus einem JScript-Objekt:

```
var Kette = new String("Hi");
alert( (Kette.constructor == String)); // liefert "true"
alert( (Kette.constructor == "String")); // liefert "false"
```

Beispiel 2 für Ableitung anhand privaten Konstruktors:

```
function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion(); // bewirkt Ausführung von TestFunktion() also auch von alert()

// ZeigerAufFunktion(); // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
//                       // da keine Ableitung vom JScript-Objekt Function
// Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration erkannt

alert(ZeigerAufFunktion.constructor == TestFunktion); // true
alert(TestFunktion.constructor == TestFunktion); // false
```

<code>.propertyIsEnumerable</code>	prüfen ob Stringwert in einem Objekt als Menge von String-Elementen enthalten ist und ob das Objekt mit der Anweisung <code>for in</code> verarbeitet werden kann Syntax: [var Wert =] object.propertyIsEnumerable(zeiger_auf_stringvariable_oder_stringwert)
	object Zeiger auf Instanz der Menge
	Wert true, so enthalten und per for in verarbeitbar false, nicht enthalten

Beispiel:

```
var Feld = new Array("Apfel", "Banane", "Zitrone");
alert( (Feld.propertyIsEnumerable("Apfel")); // true
```

<code>.prototype</code>	Zeiger auf den Prototyp-Bereich im Objekt, der per Prototyping erweitert wird Objekt ist entweder Script-Objekt oder bereits instanziiertes Objekt: innerhalb eines Konstruktors ist <code>.prototype</code> nicht zu kodieren Prototyping eines Objektes verändert den Umfang der Standard-Methoden und -Eigenschaften zum Objekt zur Laufzeit der Scriptmaschine. Script-Objekte können nur erweitert werden. Für private Objekte, die per <code>new</code> -Anweisung mit privatem Konstruktor erzeugt wurden, wird leider nicht die Eigenschaft <code>.prototype</code> erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft <code>.prototype</code> . Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft <code>.prototype</code> . siehe auch <code>.isPrototypeOf()</code> und <code>.hasOwnProperty()</code> Syntax: object.prototype.eigenschaft object.prototype.methode
	object Bezeichner des Objektes (nicht in " " bzw. ' ' kodieren) Zeiger auf instanziiertes Objekt

Beispiel für Erweiterung des Script-Objektes `Array`, das die Eigenschaft `.prototype` besitzt:



```

function MaximumErmittleIn( )
{
    var max = this[0];    // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                        // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmittleIn;    // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6);    // 6 numerische Elemente

// neue Methode des JScript-Objektes Array aufrufen
alert(Feld.NeueArrayMethode());

```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
Es wird die Eigenschaft .prototype **nicht** erzeugt !

```

<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
    this.nummer = nummer;
    this.plz = plz;
    this.ort = ort;
    this.methode = methode;
}

person = new datenstruktur_erzeugen
(
    "Erika",                // Vorname
    "Mustermann",          // Nachname
    "Musterstrasse",       // Strasse
    "1",                   // Nummer
    "10000",               // PLZ
    "Musterstadt",         // Ort
    datenstruktur_ausgeben // ohne () kodieren
);

// alternativ auch kodierbar:

```



```
// var person = {    vorname:"Erika",        // Vorname
//                  nachname:"Mustermann",    // Nachname
//                  strasse:"Musterstrasse",    // Strasse
//                  nummer:"1",               // Nummer
//                  plz:"10000",               // PLZ
//                  ort:"Musterstadt"         // Ort
//                  methode:datenstruktur_ausgeben// Ausgabemethode ohne () kodieren
//                  };
```

// Achtung: Eigenschaft .prototype wird leider **nicht** (automatisch) erzeugt und ist somit nicht anwendbar !

```
alert(person.vorname + "\n" + person.methode);    // person.methode ohne () kodieren !
// -->
</SCRIPT>
</HTML>
```

Standard-Methoden aller Objekte in JScript:

Boolean() konvertiert einen Wert nach Boolean

Syntax:

```
[ var Zeiger = ] Boolean(Wert)
```

Wert true oder false oder beliebiger Wert
wenn 0, -0, +0, null, false, NaN, undefined oder Leerkette ""
so wird false verwendet
sonst wird true verwendet
z.B. "false" wird als true verwendet

decodeURI() dekodiert einen Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde
ersetzt die Methode unescape() welche deprecated ist
ab IE 5.5

Syntax:

```
[ var Zeiger = ] decodeURI(Kette)
```

Kette encoded Uniform Resource Identifier, der mit der Methode encodeURI ()
erzeugt wurde

Zeiger auf dekodierten String

Beispiel:

```
alert(decodeURI("My%20phone%20 #%20is%20123-456-7890"));
// erzeugt "My phone # is 123-456-7890"
```

decodeURIComponent() dekodiert eine Komponente einer Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde

ab IE 5.5

Syntax:

```
[ var Zeiger = ] decodeURIComponent(Kette)
```

Kette encoded Uniform Ressource Identifier, der mit der Methode
encodeURIComponent() erzeugt wurde

Zeiger auf dekodierten String

encodeURIComponent() kodiert einen String als kompletten Uniform Ressource Identifier (URI):

Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx

Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen

```
., / ? : @ & = + $ - _ . ! ~ * ' ( ) #
Buchstaben
Ziffern
```

ersetzt die Methode escape() welche deprecated ist

ab IE 5.5

Syntax:

```
[ var Zeiger = ] encodeURIComponent(Kette)
```

Kette URI, der kodiert wird

Zeiger auf kodierten kompletten Uniform Ressource Identifier (URI)

Beispiel:

```
alert(encodeURIComponent("My phone # is 123-456-7890"));
// erzeugt "My%20phone%20 #%20is%20123-456-7890"
```



encodeURIComponent() kodiert einen Teil-String aus einem kompletten Uniform Ressource Identifier (URI):
 Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx
 Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen

. , / ? : @ & = + \$ - _ . ! ~ * ' () #
 Buchstaben
 Ziffern

ab IE 5.5

Syntax:

[var Zeiger =] encodeURIComponent (Kette)

Kette Teil-String des URI

Zeiger auf kodierten Teil-String des Uniform Ressource Identifier (URI)

escape() kodiert einen String oder ein Literal in das Unicode-Format
 ab Javascript 1.5 (Netscape 6.x) deprecated und zu ersetzen durch die Methoden
 encodeURIComponent() und encodeURIComponent()
 Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx
 Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen

. , / ? : @ & = + \$ - _ . ! ~ * ' () #
 Buchstaben
 Ziffern

URI (Uniform Resource Identifiers) werden nicht kodiert
 UTF-8 Zeichen: Teil des Unicode (0 bis 255)

Syntax:

[var Zeiger =] escape(Kette)

Kette String

Zeiger auf kodierten String

Beispiel für Einbindung einer Suchmaschine:

```
var markierter_text=document.selection.createRange().text;
// oder var markierter_text=prompt('Suchbegriff: ');
var suchmaschinen_url='http:// .....';
var suchmaschinen_parameter='.....';

if (markierter_text)
{location.HREF=suchmaschinen_url + suchmaschinen_parameter + escape(markierter_text);}
else
{location.HREF=suchmaschine_url; }
```

Beispiel für altavista: suchmaschinen_url 'http://altavista.de/'
 suchmaschinen_parameter 'cgi-bin/query?pg=q&what=web&q='

eval() parst einen String aus JavaScript-Anweisungen (kein HTML-Code) und führt diese sofort aus
 JavaScript-Anweisungen
 dürfen keine Referenz auf sich selbst haben (sonst endlose Rekursion !)
 können Referenzen auf existierende Objekte (Variablen) und deren Objektmethoden wie
 Objekteigenschaften enthalten z.B. für Wertzuweisung etc.
 Objekte (Variablen) erzeugen und instanzieren
 ab JavaScript 1.5 auch eval als Anweisung enthalten (Achtung: endlose Rekursionen
 vermeiden !)

Syntax:

[var Zeiger =] eval(Kette)

Kette String mit JavaScript-Anweisungen

Zeiger auf Ausdrucksergebnis
 nur erzeugt, wenn JavaScript-Anweisungen einen Ausdruck als letzten
 Teil oder nur einen Ausdruck in der Kette enthalten
 Es ist zu empfehlen, anstelle des Zeigers die Wertzuweisung auf eine
 Variable direkt innerhalb der JavaScript-Anweisungen zu
 realisieren.

Beispiele:

```
var Kette = eval("new String('2+2')"); // liefert Zeiger auf den String '2+2'
eval("var Kette = new String('2+2')"); // liefert nichts
// Kette hat den Wert '2+2'
```



```
eval("var Kette = new String('2+2'); alert(Kette);"); // liefert nichts, da alert nichts liefert
// Kette hat den Wert '2+2'

var StringLiteral = "2 + 2";
eval(StringLiteral) // liefert Zeiger auf numerische Variable
// mit Wert 4

var Wert = eval("2+2"); // liefert Zeiger auf numerische Variable
// mit Wert 4

var StringObjekt = new String("2 + 2");
eval(StringObjekt) // liefert Zeiger auf Kette "2 + 2"
```

Beispiel:

```
var w = 2;
var x = 39;
var y = "42";
var z = "42"
eval("w + x + 1"); // liefert Zeiger auf numerische Variable
// mit Wert 42

eval(y); // liefert Zeiger auf numerische Variable
// mit Wert 42
// liefert nicht Zeiger auf y, da der String
// "42" ausgewertet wird

eval(z); // liefert Zeiger auf String-Variable
// mit Wert '42'
// liefert nicht Zeiger auf z, da der String
// "42" ausgewertet wird
```

Beispiel:

```
var Kette = "var x=5;"
+ "if (x == 5)"
+ "{"
+ "alert('Meldung aus eval\nx ist 5');" // \n ist Zeilenumbruch-Zeichen
+ "x = 42;"
+ "}"
+ "else"
+ "{"
+ "x = 0;"
+ "}";

var Kette1 = "alert('Meldung aus document.write\nx ist ' + eval(Kette));"
document.write(Kette1);
```

Beispiel für Ersatz von eval() durch die Eigenschaft .innerHTML (nur IE):

```
<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette; // wird sofort geparkt, also IMG-Tag ausgeführt
// ( anstelle von eval()
// bzw. document.write() )

    ID_IMG.src="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>
```

Beispiel:

```
if (VarUserAgent != "")
{
    for ( var i=0; i < 10; i++)
    {
```



```

        eval(
            'if (VarUserAgent.indexOf("'" + i + "'.") != -1)'
            + '{VarBrowser_Version_Haupt = ' + i + '};'
        );
    }
}

```

Beispiel:

```

function DatumHolen(Zeiger)
{
    var Kette = "Heute ist: ";
    Kette += Zeiger.getDate() + "/";
    Kette += (Zeiger.getMonth() + 1) + "/";
    Kette += Zeiger.getYear();

    return(Kette);
}

var Kette="Date";

eval(
    "var Jetzt = new " + Kette + "();" // zur Laufzeit erzeugen
    "alert(DatumHolen(Jetzt));" // Parameter ist zur Laufzeit bekannt
);

```

.hasOwnProperty() prüfen ob zum Objekt eine Eigenschaft vorhanden ist, jedoch leider **nicht** aus Prototyping einer per new erzeugten Instanz des JScript-Objektes

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft **.prototype** erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft **.prototype**.

siehe auch **.prototype**
 Syntax:

```
[ var Wert = ] object.prototype.hasOwnProperty(kette)
```

kette String
 Bezeichner der Eigenschaft

objekt auf per new erzeugte Instanz oder Bezeichner eines JScript-Objektes

Wert true, so Eigenschaft vorhanden
 false, so Eigenschaft nicht vorhanden

Beispiel:

```

function MaximumErmittleIn( )
{
    var max = this[0]; // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                      // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmittleIn; // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6); // 6 numerische Elemente
// Es wird die Eigenschaft .prototype erzeugt
alert( (Feld.prototype.hasOwnProperty("NeueArrayMethode")); // leider false
alert( (Array.prototype.hasOwnProperty("NeueArrayMethode ")); // true

```

isFinite() ermittelt, ob Wert einer Instanz numerisch endlich ist
 Syntax:

```
[ var Wert = ] isFinite(Zeiger);
```

Zeiger auf Instanz vom Number-Objekt oder Ausdrucksergebnis
 auch Number.POSITIVE_INFINITY
 Number.NEGATIVE_INFINITY
 Number.NaN



Wert true, so Wert der Instanz ist endlich
false, so Wert der Instanz ist unendlich
immer bei
Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY
Number.NaN

isNaN() ermittelt, ob Wert einer Instanz nicht numerisch ist
Syntax:

[var Wert =] isNaN(Zeiger);

Zeiger auf Instanz vom Number-Objekt oder Ausdrucksergebnis
auch Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY
Number.NaN

Wert true, so Wert der Instanz ist nicht numerisch
immer bei Number.NaN
false, so Wert der Instanz ist numerisch
immer bei
Number.POSITIVE_INFINITY
Number.NEGATIVE_INFINITY

Beispiele:

```
var Kette = "10.23";
var Wert = 10.23;

var Ergebnis1 = parseFloat(Kette);
var Ergebnis2 = floatValue = parseFloat(Wert);

alert(isNaN(Ergebnis1));
alert(isNaN(Ergebnis2));
```

Hinweis: Methoden parseFloat und parseInt liefern NaN, wenn Parameter nicht numerisch ist

Bsp: var zahl = parseFloat("text"); if NaN() liefert true

.isPrototypeOf() prüfen ob Objekt per new erzeugt wurde, also eine Instanz eines anderen JScript-Objektes ist
Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht**
die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors
erfolgen und nicht nachträglich per Eigenschaft .prototype .
siehe auch .prototype
Syntax:

[var Wert =] object.prototype.isPrototypeOf(zeiger)

zeiger auf per new erzeugte Instanz

object Zeiger auf JScript-Objekt, dessen Bezeichner als
Konstruktor in der new Anweisung verwendet wurde

Wert true, so Objekt ist Instanz eines JScript-Objektes
false, so keine Instanz eines JScript-Objektes
oder Zeiger auf per new erzeugte Instanz ist ungültig

Beispiel:

```
var Variable = new RegExp();
alert( (RegExp.prototype.isPrototypeOf(Variable)); // true.
```

Number() konvertiert eine Instanz zu einem numerischen Wert (Number-Objekt-Wert)
Syntax:

[var Wert =] Number(Zeiger)

Zeiger 1 auf zu konvertierende Instanz
auch Date-Objekt

Wert wenn Zeiger 1 ein Date-Objekt ist, so Wert ist die Anzahl der
Millisekunden seit dem 1.1.1970 0 Uhr Weltzeit
(UTC), wobei Weltzeit genau GMT entspricht
wenn Datum vor dem 1.1.1970 0 Uhr Weltzeit so Wert negativ
NaN wenn Konvertierung nicht möglich ist

Beispiel:

```
var DatumJetzt = new Date();
alert (Number(DatumJetzt));
```



`parseFloat()` String oder Literal parsen und nach Floating-point umwandeln können enthalten

- Vorzeichen + und -
- Ziffern 0 bis 9
- Dezimalkomma als Punkt
- e oder E
- Blanks (werden ignoriert)

falls andere Zeichen enthalten sind, gilt:
String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann konvertiert

Erfolg der Konvertierung ist per Methode `isNaN()` zu prüfen

Syntax:

```
[ var Wert = ] parseFloat(Kette)
```

Kette String oder Literal

Wert Floating-point
NaN wenn String bzw. Literal bereits mit erstem Zeichen fehlerhaft

Beispiele:

gültiges Literal

```
parseFloat("3.14")
parseFloat("314e-2")
parseFloat("0.0314E+2")
```

gültiger String

```
var x = "3.14"
parseFloat(x)
```

ungültiges Literal

```
parseFloat("FF2")
```

```
alert(parseFloat("17.50 DM"));
```

`parseInt()` String oder Literal parsen und nach Integer umwandeln können enthalten

- Vorzeichen + und -
- Ziffern 0 bis 9, aber keine Vornull(en)
- Blanks (werden ignoriert)
- eventuelle Buchstaben A bis F

falls andere Zeichen enthalten sind, gilt:
String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann konvertiert

Erfolg der Konvertierung ist per Methode `isNaN()` zu prüfen

Syntax:

```
[ var Wert2 = ] parseInt(Kette[, Wert1])
```

Kette String oder Literal
muss passend zu Wert1 kodiert sein
Hinweis: oktale Escape-Sequenzen sind in ab Javascript 1.5 im Netscape 6.x deprecated

Wert1 Integer
Zahlenbasis

- 10 für dezimal
- 8 für oktal
- 16 für hexadezimal

(Wert1 kann Buchstaben A bis F enthalten)

Standard: 0

Wert2 Integer
NaN wenn String bzw. Literal bereits mit erstem Zeichen fehlerhaft
wenn Wert1 0 ist, so
wenn Kette beginnt mit
"0x", so hexadezimale Konvertierung
"0" und nicht "0x", so oktale Konvertierung
nicht mit "0" oder "0x", so dezimale Konvertierung



Beispiel:

gültige Literale:

```
parseInt("F", 16)
parseInt("17", 8)
parseInt("15", 10)
parseInt(15.99, 10)
parseInt("FXX123", 16)
parseInt("1111", 2)
parseInt("15*3", 10)
parseInt("17")
parseInt("0x7", 16)
parseInt("0x7")
parseInt("0")
parseInt("0x11", 16)
parseInt("0x11", 0)
parseInt("0x11")
```

ungültige Literale:

```
parseInt("F")
parseInt("Hello", 8)
parseInt("0x7", 10)
parseInt("FFF", 10)
parseInt("002")
```

String()

konvertiert eine Instanz zu String
ist identisch mit Methode .toString()
Syntax:

```
[ var Kette = ] String(Zeiger)
```

Zeiger auch auf Date-Objekt

Kette wenn Zeiger auf Date-Objekt, so Datum als String geliefert (je nach
Ländereinstellung des Windows)

Beispiel: Thu Aug 18 04:37:43 GMT-0700 (Pacific Daylight Time) 1983

.toLocaleString()

Wert eines Objektes in System-lokale Einstellungen umwandeln
System-lokale Einstellung z.B. Ländereinstellung, Uhrzeitformat
Konvertierung: Analog wie z.B. bei Excel, das die lokalen Einstellungen ausliest.
nur anwenden für Anzeige des konvertierten Wertes:

Berechnungen mit dem Wert immer unkonvertiert vollziehen, da sämtliche
Berechnungsfunktionen **nur** das interne, also unkonvertierte
Format kennen (**nicht** analog zu Excel)

Wenn das Objekt ein Script-Objekt Array ist, also Feldelemente in lokale Einstellungen konvertiert werden
sollen, dann wird ein String geliefert, der die Feldelemente in der Reihenfolge im Feld
und diese getrennt durch **dasjenige** Trenner-Zeichen laut **lokale** Einstellungen enthält.

Wenn das Objekt eine Script-Objekt Date ist, so wird der Wert von dem Objekt in den lokalen
Datumeinstellungen geliefert. Dabei sind nur Jahreszahlen von 1601 bis 9999 zulässig.
Andere Jahresangaben werden nicht konvertiert.

Wenn das Objekt ein Script-Objekt Number ist, so werden die lokalen Einstellungen zum Zahlenformat
geliefert.

Wenn das Objekt ein Script-Objekt Object ist, so werden nur dann lokale Einstellungen berücksichtigt,
insoweit das Objekt diese berücksichtigen kann. Ein String wird immer geliefert.

Syntax:

```
[ var Kette = ] object. toLocaleString()
```

object Referenz auf den Wert eines Objektes

Kette String

Beispiel für Konvertierung eines Feld mit Gleitkomma-Werten:

```
var Feld = new Array(6);
var Wert = 3,201,300.20; // in JScript ist das Dezimalkomma der Punkt
                        // der Tausendertrenner das Komma

// Feld füllen
for( var i = 0; i < 7; i++)
{
    Wert +1 = 10;
    Feld [i] = Wert;
}
```



```
alert(Feld.toLocaleString());
```

Beispiel für Konvertierung eines Datums:

```
var Jetzt = new Date();
alert(Jetzt.toLocaleString());
```

.toString()

Wert eines Objektes in einen String umwandeln
 wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma getrennt geliefert
 wenn Objekt ein Script-Objekt Boolean ist, dann "true" bzw "false" geliefert (Kleinschreibung !)
 wenn Objekt ein Script-Objekt Error ist, dann die Fehlermeldung geliefert (Objekt-Wert ist der Fehlercode)
 wenn Objekt ein Script-Objekt Function ist, dann Quellcode der Funktion geliefert (inklusive Funktionskopf)
 wenn Objekt ein Script-Objekt Object ist, so wird geliefert:
 "**[object** objectname]"
 mit objectname als konkreter Bezeichner der Objektklasse bzw. des Objektes
 fettgedrucktes wird so geliefert wie angegeben
 wenn der Bezug vor .toString() ein Wert laut ID-Attribut oder NAME-Attribut ist, dann wird die Objektklasse geliefert

Syntax:

```
[ var Kette = ] object.toString([Wert])
```

Wert	nur kodierbar bei numerischen Wert (Objekt ist Number-Objekt)
Basis des Wertes	
z.B.	2 für dual (binär)
	10 für dezimal
	16 für hexadezimal
object	Referenz auf den Wert eines Objektes
Kette	String

Beispiel 1:

```
var Wert = 33,33;
alert(Wert.toString(2) + ' ' + Wert.toString(16) + ' ' + Wert.toString(10));
```

Beispiel 2:

```
<BUTTON ID="ID_Button" .... > .... </BUTTON>
ID_Button.toString() liefert "button"
```

unescape

ab Javascript 1.5 (Netscape 6.x) deprecated und zu ersetzen durch die Methoden
 decodeURI() und decodeURIComponent()
 dekodiert einen mit der Methode escape() kodierten String (siehe escape())

Syntax:

```
[ var Zeiger = ] unescape(Kette)
```

Kette	String aus escape()
Zeiger	auf dekodierten String

.valueOf()

Wert eines JScript-Objektes bzw. Instanz eines JScript-Objektes ermitteln
 nicht bei Script-Objekt Math und Script-Objekt Error (auch nicht bei Instanzen dieser Objekte)
 Wert ist im Datentyp des Objektes, aber:
 wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma getrennt geliefert (identisch in der Wirkung mit .toString() und der Array-Methode join())
 wenn Objekt ein Script-Objekt Boolean ist, dann true bzw false geliefert (kein String !)
 wenn Objekt ein Script-Objekt Date ist, dann Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr geliefert
 wenn Objekt ein Script-Objekt Function ist, dann Zeiger auf Funktion geliefert
 wenn Objekt ein Script-Objekt Number ist, dann numerischen Wert geliefert
 wenn Objekt ein Script-Objekt Object ist, so Zeiger geliefert
 siehe auch .toLocaleString() und .toString()

Syntax:

```
[ var Wert = ] object.valueOf( );
```

Wert	typengerecht zum JScript-Objekt
object	nur JScript-Objekt bzw. Instanz davon



4.2. In Javascript vordefinierte Objekte (Script-Objekte)

(Objektklassen oder Objekttypen)

Es können alle Objekte aus Javascript/JScript mit deren Eigenschaften und Methoden in sinnvoller Kombination mit den zum Browser vordefinierten Objekten verwendet werden. Die objekt-übergreifenden, also objekt-unabhängigen Methoden, sind alle nutzbar aber z.T. in den zum Browser vordefinierten Objekten abgewandelt implementiert.

Die in Script definierten Objekte sind z.B.

- Objekt arguments
- Objekt Array
- Objekt Boolean
- Objekt Date
- Objekt Enumerator
- Objekt Error
- Objekt Function
- Objekt Math
- Objekt Number
- Objekt Object (nicht Objekt object des Internet Explorer für das HTML-Tag OBJECT)
- Objekt RegExp
- Objekt String
- Objekt var

Die Bezeichner dieser Objekte werden auch **Objekttypen oder Objektklassen** benannt. Letztere Begriffe finden **nur** in der Ableitung eines vordefinierten Objektes Anwendung: Von den vordefinierten Objekten sind Instanzen per Anweisung new erzeugbar (siehe dort), wobei als Konstruktor der Bezeichner des Objektes verwendet wird. Es gibt Script-Objekte, die sich nicht ableiten lassen (z.B. Math Script-Objekt), da der Hersteller verhindern will, dass Veränderungen am Objekt vorgenommen werden.

Man beachte: Es können auch rein private Objekte erzeugt werden, die nicht von einem vordefinierten Objekt abstammen.

Prototyping einer Ableitung ist nur durch die Verwendung der Eigenschaft .prototype möglich, allerdings das auch nur dann, wenn es sich um Objekte handelt, die aus den **vordefinierten** Script-Objekten per new-Anweisung abgeleitet wurden, denn **nur dann** wird die Eigenschaft .prototype überhaupt erzeugt.

Die Art und der Umfang der Implementation der vordefinierten Objekte in die Scriptmaschine hängt vom Hersteller ab.

4.2.1. arguments Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.
nur verfügbar während Abarbeitung der zugehörigen Funktion und von dieser instanziiert
immer funktionslokal

hat Feldcharakter, das alle Argumente der Funktion referenziert:

jedes Argument muss mit Parameter versorgt werden
Anzahl der Elemente in Argumenten- und Parameterliste muss identisch sein
Alle Parameter werden in der Listenfolge von links nach rechts in arguments abgelegt
mit Index ab 0 und aufsteigend
Anzahl der Parameter laut arguments.length

Bsp: function test(arg1, arg2, arg3) {.....}

test(10, 20, 30);

10	entspricht test.arguments[0] für arg1
20	entspricht test.arguments[1] für arg2
30	entspricht test.arguments[2] für arg3

selbst Eigenschaft der Funktion ist

siehe Script-Objekt Function und Anweisung function

Syntax:

```
[ var Zeiger = ] [funktions_bezeichner].arguments
[ var ZeigerAufElement = ] [funktions_bezeichner].arguments [Index]
[ var ZeigerAufElement = ] [funktions_bezeichner].arguments.0 bis .n
```

funktions_bezeichner nur dann zu kodieren, wenn arguments innerhalb einer Funktionsverschachtelung verwendet wird, um deren Argumentenlisten zu trennen
Funktionsbezeichner laut Deklaration und Aufruf der Funktion

Index Integer, ab 0
muss in [] kodiert sein



.0 bis .n Eigenschaften von arguments

Zeiger ist null, wenn keine Argumente vorhanden

ZeigerAufElement ist null, wenn Feldelement nicht vorhanden

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

```
var GlobalesFeld = new Array();

function FunktionsBezeichner()
{
    // Argumenteliste der Funktion lokal zur Funktion referenzieren
    var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

    // Anzahl der Argumente
    var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

    if ( ArgumenteAnzahl > 0 )
    {
        // Arrgumentenliste auslesen
        for (var i = 0; i < ArgumenteAnzahl; i++)
        {GlobalesFeld[i] = ArgumentenListeAlsFeld[i];}
    }

    // hier die weiteren Anweisungen der Funktion
}
```

Eigenschaften:

.0 bis .n Wert des Argumentes mit Index 0 ... bzw. n im Script-Objekt arguments als Eigenschaft eines Funktionsobjektes (siehe Script-Objekt Function und Anweisung function)

.0 entspricht auch funktions_bezeichner.arguments[0]

.n entspricht auch funktions_bezeichner.arguments[n]

n von 0 bis beliebig ganzzahlig-numerische Ziffernfolge ohne Vornull

ist die Position ab 0 innerhalb der Argumentenliste von links nach rechts

Parameterliste von links nach rechts

Hinweis: Argumentenliste und Parameterliste mit gleicher Elementanzahl, denn **jedes Argument** muss mit Wert versorgt werden

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

```
var GlobalesFeld = new Array();

function FunktionsBezeichner()
{
    // Argumenteliste der Funktion lokal zur Funktion referenzieren
    var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

    // Anzahl der Argumente
    var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

    if ( ArgumenteAnzahl > 0 )
    {
        // Arrgumentenliste auslesen
        GlobalesFeld[0] = ArgumentenListeAlsFeld.0;
        GlobalesFeld[1] = ArgumentenListeAlsFeld.1;
    }

    // hier die weiteren Anweisungen der Funktion
}
```

.callee Zeiger auf den Funktionsrumpf

z.B. verwenden, wenn

Funktion ohne Funktionsbezeichner erzeugt wurde

für Kodierung einer Rekursion ohne den Funktionsbezeichner aber **mit Argumentenliste**

(falls Argumentenliste vorhanden ist)

Beispiel:

```
function Test(n)
{
    if (n <= 0)
```



```

        {return 1;}
        else
        {return (n * arguments.callee(n - 1));}    // Rekursion per arguments.callee(n - 1)
    }

    alert(Test(3));

```

.length Anzahl der Argumente im Script-Objekt arguments als Eigenschaft eines Funktionsobjektes
Wert ist identisch mit dem Wert der Eigenschaft **.length** des Script-Objektes Function

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

```

var GlobalesFeld = new Array();

function FunktionsBezeichner()
{

    // Argumenteliste der Funktion lokal zur Funktion referenzieren
    var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

    // Anzahl der Argumente
    var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

    if ( ArgumenteAnzahl > 0 )
    {
        // Argumentenliste auslesen
        for (var i = 0; i < ArgumenteAnzahl; i++)
        { GlobalesFeld[i] = ArgumentenListeAlsFeld[i]; }
    }

    // hier die weiteren Anweisungen der Funktion
}

```

Methoden:

keine

4.2.2. **Array Script-Objekt**

Dieses Objekt ist eine Komponente der Scriptsprache.

Sammlung von Werten mit beliebigen Datentypen und Referenzierung der Werte per Integer-Index als Position eines Wertes in der Sammlung

frei gestaltbar

siehe auch Basis-Datenstruktur array

4.2.2.1. **Array Script-Objekt mit Elemente eines beliebigen Datentyps**

Syntax:

Syntax für Felderzeugung: Die new-Anweisung kann zwar entfallen, sollte aber kodiert werden.

```

[ var Zeiger = ] new Array()                // leer, also 0 Elemente
                                           // ermöglicht mehrdimensionales Feld

[ var Zeiger = ] new Array(wert)

        wert      Anzahl der Feldelemente
                   Integer ab 0

[ var Zeiger = ] new Array(werte_liste)
[ var Zeiger = ] new Array[werte_liste]    // ab Javascript 1.2

        werte_liste      initialisiert Feldelemente ab Index 0 aufsteigend mit den Werten laut Liste von links
                           nach rechts
                           kommagetrennte Werte z.B.
                           Literal
                           Wert laut Referenz
                           Wert aus Funktionsaufruf
                           Wert als Ergebnis eines Ausdrucks
                           [] ist zu kodieren !

```

Syntax für Feldfüllung ohne new-Anweisung aber anhand Basis-Datenstruktur array:

```

Zeiger = [ [ liste_1 ] [ ..... , [ liste_n ] ]      für mehrdimensionales Feld
Zeiger = [ liste ]                                für eindimensionales Feld

```

Zeiger **instanziiertes** Objekt der Objektklasse Array, also Ableitung vom Script-Objekt Array



```
per var Zeiger = new Array()
```

[] bedeutet hier **nicht** optional !!

[liste_1] bis [liste_n]

Liste aus kommasetrennten Elementen

Liste_x stellt ein symbolisches Feld da (x ist 1 n)

liste Liste aus kommasetrennten Elementen

Listenelement: kann Ausdruck sein, der Wert liefern muss
kann entfallen, aber das trennende Komma muss kodiert werden
Folge der Listenelemente entspricht Initialisierungsfolge des Feldes
pro Komma eine automatische Indexerhöhung
wenn Element nicht kodiert, so wird das Feldelement
zum Index nicht initialisiert ---> Feldelement
hat keinen Datentyp

Syntax für Feldfüllung aus Literalen:

siehe weiter unten

Zugriff:

Zeiger[Index].eigenschaft

Zeiger[Index].methode()

Zeiger[Index] = wert

Index Integer ab 0
oder String (in " " bzw. ' ' kodieren)
muss in [] kodiert sein

Wert füllt Feldelement und bestimmt den Datentyp des Feldelementes: Feldelemente können verschiedentypig sein

Beispiele:

Beispiele für Füllen eines instanziierten und leeren Feldes mit Namen "Feld":

Feld = [1,2,3]; Element an Index 0 hat Wert 1
Element an Index 1 hat Wert 2
Element an Index 2 hat Wert 3

Anzahl der Elemente 3

Feld = [1,...,5]; Initialisiert wird **nur** mit Index 0 auf Wert 1
mit Index 4 auf Wert 5
Anzahl der Elemente 2

Feld = [["Name", "Tom", "Tim", "Teo"], ["Alter", 6, 5, 4]];

Feld[0] = 10;
Feld[99]=100; // Feldlänge ist 2 !!!

Beispiel für Erzeugung eines 2-dimensionalen Feldes:

```
function erzeuge_zwei_dim_feld(anzahl_spalten, anzahl_zeilen)
{
    // Spaltenfeld erzeugen
    this.spalten_feld= new Array(anzahl_spalten);

    // Zeilenfeld pro Spalte erzeugen
    for ( var spalte=0; spalte < anzahl_spalten; spalte++)
    { this[spalte] = new Array(anzahl_zeilen); } //Zeilen-Felder
}

var zwei_dim_tabelle= new erzeuge_zwei_dim_feld(10,7); // 10 Spalten zu je 7 Zeilen
....
zwei_dim_tabelle[10,7]=22; // Spalte 10: 7Zeile: mit 22 belegen

.....
delete zwei_dim_tabelle[10,7]; // löscht in Spalte 10 die 7. Zeile
```

Beispiel für Erzeugung eines 2-dimensionalen Feldes:



```

var Feld = new Array();
var AnzahlUnterfelder = 4;
var AnzahlElementeImUnterfeld = 4;

// Unterfelder erzeugen
for (    var UnterfeldZahler=0;
        UnterfeldZahler < AnzahlUnterfelder;
        UnterfeldZahler++)
{
    // jedes Unterfeld als Feld aus Elementen erzeugen
    Feld[UnterfeldZahler] = new Array();

    // Unterfeld elementeweise füllen mit Wert "[UnterfeldZahler,Unterfeld_Elemente_Zahler]"
    // wobei UnterfeldZahler und Unterfeld_Elemente_Zahler für die Indexwerte stehen

    for (    Unterfeld_Elemente_Zahler=0;
            Unterfeld_Elemente_Zahler < AnzahlElementeImUnterfeld;
            Unterfeld_Elemente_Zahler++)
    {
        FeldGesamt[UnterfeldZahler][Unterfeld_Elemente_Zahler] =
            "["
            + UnterfeldZahler
            + ","
            + Unterfeld_Elemente_Zahler
            + "];"
    }
}

```

Beispiel für Erzeugung eines beliebig-dimensionalen Feldes:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function mehr_dim_array_erzeugen( ein_dim_array_anzahl, ein_dim_array_elemente_anzahl)
    {
        // ist Konstruktor für Erzeugung eines mehrdimensionalen Arrays
        // als Menge von eindimensionalen Arrays

        for (    var ein_dim_array_zahler = 0;
                ein_dim_array_zahler < ein_dim_array_anzahl;
                ein_dim_array_zahler++)
        {
            this[ein_dim_array_zahler] = new Array(ein_dim_array_elemente_anzahl);
            // ein_dim_array_zahler immer ab 1
            // je ein eindimensionales Array erzeugen und zu this zuordnen
            // this ist Zeiger des Konstruktors auf das per new
            // anzulegende mehrdimensionale Feld
        }

        this.breite = ein_dim_array_anzahl;
        this.hoehe = ein_dim_array_elemente_anzahl;
    }
// -->
</SCRIPT>
</HEAD>

<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!--
    zwei_dim_array = new mehr_dim_array_erzeugen (2, 3);
    // zwei Arrays zu je 3 Feldelementen,
    // zweidimensionale Matrix mit
    // 2 Zeilen zu je 3 Spalten
    // erzeugen
    // Anzahl immer ab 1

```



```

zwei_dim_array[1][2] = 17;
// letztes eindimensionale Feld in dessen
// Element 2 (vorletztes) mit Wert 17 initialisieren
// Indexe immer ab 0
document.write("Feld 1-2: " + zwei_dim_array[1][2]);
// -->
</SCRIPT>
</BODY>
</HTML>

```

Beispiel zur Konvertierung von Hexaziffern nach numerisch:

```

var hexaziffern_feld = [
    // anstelle von new Array()
    "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
    "A", "B", "C", "D", "E", "F"
];

function dezimalwert_zu_hexaziffern(dezimal_wert)
{
    var high = Math.floor(dezimal_wert / 16);
    var low = Math.floor(dezimal_wert - (high * 16));

    return(hexaziffern_feld[high] + hexaziffern_feld[low]);
}

```

Beispiele zu push und pop:

ab IE 5.5 bzw. NS 6.x existiert die Objektmethode .push() und .pop()

```

var names = new Array("Tom", "Mark", "Bart", "John");
names.push("Jim", "Richard", "Tim");
alert(names);
var last = names.pop();
alert("Last = " + last + " Other = " + names);

```

vor IE 5.5 bzw. NS 6.x muss push und pop durch den Programmierer per Prototyping implementiert werden

Beispiel für push:

```

function push()
{
    var sub = this.length;
    for (var i = 0; i < push.arguments.length; ++i)
    {
        this[sub] = push.arguments[i];
        sub++;
    }
}

var names = new Array("Tom", "Mark", "Bart", "John");
// Es wird die Eigenschaft .prototype erzeugt

// Prototyping der Instanz names
names.prototype.push = push;

// auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.push = push;

```

Beispiel für pop:

```

function pop()
{
    var lastElement = this[this.length - 1];
    this.length--;
    return lastElement;
}

var names = new Array("Tom", "Mark", "Bart", "John");
// Es wird die Eigenschaft .prototype erzeugt

// Prototyping der Instanz names

```



```
names.prototype.pop = pop;
```

```
// Auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.pop = pop;
```

Beispiele zu shift und unshift:

Beispiel für unshift-Methode bei Browsern nicht ab IE5.5 bzw. nicht ab NS6.x:

```
function unshift()
{
    var i = unshift.arguments.length;
    for (var j = this.length - 1; j >= 0; --j)
    { this[j + i] = this[j]; }

    for (j = 0; j < i; ++j)
    { this[j] = unshift.arguments[j]; }
}

Array.prototype.unshift = unshift;
```

Beispiel für shift -Methode ab IE5.5 bzw. NS 6.x:

```
var line = new Array("aaa", "bbb", "ccc", "ddd", "eee");
alert("first = " + line.shift() + " Other = " + line);
```

Beispiel für unshift -Methode ab IE5.5 bzw. NS 6.x:

```
var line = new Array("ccc", "ddd", "eee");
line.unshift("aaa", "bbb");
alert(line);
```

Eigenschaften:

.length

Anzahl der Elemente in einem Feld der Objektklasse Script-Objekt Array
Maximale Anzahl von Feldelementen: 4294967295

Methoden:

.concat()

Feld (Quellfeld1) kopieren in eine neue und automatisch erzeugte Instanz (Zielfeld) und optionales Anhängen von Werten (z.B. weiteren Quellfeldern2 ...) an das Ende des Zielfeldes.

Quellfeld1 kann leer sein

Verkettung erfolgt in der Folge der Kodierung der zu verkettenden Objekte, Ausdrücke etc.:

Erstes Objekt in der Verkettung ist immer Quellfeld1.

wenn mehrere Quellfelder zu verkettet sind, dann gilt

wenn Feldelement **nicht** String **und nicht** Number ist, so

wird das Feldelemente als Zeiger kopiert bzw. verkettet

der Wert, auf den der Zeiger weist, wird nicht verwendet

bewirkt Änderung eines Wertes im Quell- bzw. Zielfeld auch die Änderung im Ziel- bzw. Quellfeld, da identische Zeigerbezüge vorliegen

wenn Feldelement String oder Number ist, so

erfolgt Wertekopierung, also keine Zeigerkopierung

bewirkt Änderung eines Wertes im Quell- bzw. Zielfeld keine Änderung im

Ziel- bzw. Quellfeld, da keine Zeigerbezüge vorliegen

Änderung der Anzahl der Feldelemente in den Quellfeldern hat keine Auswirkung auf die Anzahl der Feldelemente im Zielfeld

Verkettung von Feldern erfolgt

feldweise in der Folge der Kodierung der Quellfelder

und pro Quellfeld: elementweise in der Folge der Feldelemente

Felder haben die Objektklasse Script-Objekt Array

ab IE 5.5 und NS 6.x siehe auch .push()

Syntax:

```
[ var Zeiger = ] zeiger_auf_quell_feld.concat([werte_liste])
```

werte_liste

optional

kommagetrennte Werte

Wert kann z.B. sein

Literal

Referenz z.B. auf Feld

aus Ausdruck

aus Funktionsaufruf

Zeiger

auf die neue Instanz (Zielfeld), die automatisch erzeugt wird

Beispiel:

```
var Feld1 = new Array(0,1);
```



```
var Feld2_Quelle = new Array(2, 3);
var Wert_Quelle = 4;
var Feld3_Ziel = Feld1.concat(Feld2_Quelle, Wert_Quelle);
alert(Feld3.join()); // "0,1,2,3,4"
```

Beispiel:

```
var QuellFeld1=new Array("a","b","c");
var QuellFeld2=new Array(1,2,3);
var ZielFeld=QuellFeld1.concat(QuellFeld2) // Zielfeld enthält Wertkopien also ["a","b","c",1,2,3]
```

Beispiel:

```
var Variable1="Hallo ";
var Variable2="Du";
var Variable3="!";

var QuellFeld1=new Array(Variable1,Variable2,Variable3);
var QuellFeld2=new Array("Hallo ","Du","!");
var ZielFeld=QuellFeld1.concat(QuellFeld2);
// Zielfeld enthält Wertkopien also [      Zeiger_Auf_Variable1,
//                                     Zeiger_Auf_Variable2,
//                                     Zeiger_Auf_Variable3,
//                                     "Hallo ","Du","!"
//                                     ]
```

.join()

Feld auslesen und als String liefern, wobei jedes Feldelement mit einem freidefinierbaren

Trenner im String getrennt wird, der mit eingebaut wird

Feld elementweise nach String kopieren und als 1 gemeinsamen String liefern

Feld hat die Objektklasse Script-Objekt Array

Syntax:

```
[ var Kette1 = ] zeiger_auf_feld.join([Kette2]);
```

Kette2 optional
String
ist der Trenner als Zeichefolge
wird in Kette1 eingebaut
Standard ist "," also Komma

Kette1 String
Feldinhalt mit Elemententrennung laut Kette 1
ist Leerkette wenn mindestens ein Feldelement nicht vorhanden ist
weil undefined bzw. null

Beispiele:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.join("-")); // "0-1-2-3-4"
alert(Feld.join()); // "0,1,2,3,4"
```

```
var Feld = new Array("Wind","Rain","Fire");
var Kette = Feld.join(" + "); // Kette enthält "Wind + Rain + Fire"
```

.reverse()

Reihenfolge der Feldelemente physisch umkehren, also

<u>alt</u>	<u>neu</u>
1. Element	letztes Element
letztes Element	1. Element

Feld hat die Objektklasse Script-Objekt Array

Syntax:

```
zeiger_auf_feld.reverse()
```

liefert nichts

Beispiel:

```
var Feld1 = new Array(0,1,2,3,4); // Feld 1 ist Zeiger vor reverse
Feld1.reverse(); // Feld 2 ist Zeiger nach reverse, Feld1 ist ungültig
alert(Feld1.join()); // "4,3,2,1,0"
```

.slice()

Elementfolge aus Quellfeld als neues Feld (Zielfeld) liefern

nachträgliche Änderung der Elementanzahl im Quellfelde wirken sich nicht auf Zielfeld aus
(keine dynamische Verkettung)

wenn Quellfeld **nicht** String **und nicht** Number enthält:

neues Feld hat Feldelemente als Zeiger auf die Elemente also nicht als Wertkopie

Werte-Änderung im Quellfeld bewirkt sofortige Änderung im Zielfeld, da identische Zeiger
vorliegen

wenn Quellfeld Strings oder Number enthält:



neues Feld hat Feldelemente als Wertkopien aus dem Quellfeld

Werte-Änderung im Quellfeld bewirkt keine Änderung im Zielfeld, da nicht identische Zeiger vorliegen

Felder haben die Objektklasse Script-Objekt Array

Syntax:

```
[ var Zeiger = ] zeiger_auf_feld.slice(StartIndex[,EndeIndexPlus1])
```

StartIndex Index, ab 0 und < Länge des Feldes laut .length

EndeIndexPlus1 Integer,

wenn 0, so > StartIndex

Achtung: Element laut EndeIndexPlus1 wird nicht verwendet !

wenn < 0, so .length + EndeIndexPlus1 verwendet

Bsp.: EndeIndexPlus1 ist -1

also .length + (- 1) verwendet

Achtung: Element laut .length + EndeIndexPlus1 wird nicht verwendet !

Standard: .length (also Elemente verwendet bis

Index .length -1)

Beispiel:

```
var Feld1 = new Array(0,1,2,3,4);
var Feld2 = Feld1.slice(0,1);
alert(Feld2.join()); // "0"
var Feld3 = Feld1.slice(1);
alert(Feld3.join()); // "1,2,3,4"
```

Beispiel:

```
var QuellFeld=new Array("a","b","c")
var ZielFeld=QuellFeld.slice(0,2) // Zielfeld enthält Wertkopien also ["a","b"]
```

Beispiel:

```
var Variable1="Hallo ";
var Variable2="Du";
var Variable3="!";

var QuellFeld=new Array(Variable1,Variable2,Variable3);
var ZielFeld=QuellFeld.slice(0,-1) ; // 2 verwendet, denn Länge 3 minus 1 ist 2
// Zielfeld enthält Wertkopien also [ Zeiger_Auf_Variable1,
//                                     Zeiger_Auf_Variable2
//                                     ]
```

.sort()

Feldelemente physisch sortieren

sind zwei direkt zusammenliegende Feldelemente wertmäßig identisch, so werden deren Positionen im Feld nicht geändert

ein Feldelement mit Wert undefined landet am Feldende:

Liegen mehrere Feldelemente mit Wert undefined vor, so landen diese in der Reihenfolge vor der Sortierung am Ende des Feldes nach der Sortierung

Standardsortierung : laut ASCII-Zeichensatz und aufsteigend

Feld hat die Objektklasse Script-Objekt Array

Syntax:

```
zeiger_auf_feld.sort([Zeiger])
```

Zeiger optional

wenn nicht kodiert, so

Sortierung lexikographisch

jedes Feldelement wird dafür automatisch temporär in String umgewandelt

Bsp.: Wert 80 wird zu "80"

Wert 9 wird zu "9"

sortieren ergibt 80 und dann 9

wenn kodiert, so Zeiger auf Funktion, also

Funktionsbezeichner **ohne** () und **ohne**

Parameter kodieren

Funktion vergleicht 2 Feldelemente

Funktion muss folgenden Aufbau haben:

```
freier_bezeichner(Arg1,Arg2)
```

```
// Bezeichner der Argumente frei wählbar
```

```
// Vergleich auf <= und >= leider nicht zulässig
```

```
{
```




```

var Wert = 0;

if (Arg1 < Arg2)
{Wert--;} // muss < 0 sein
else
{
    if (Arg1 > Arg2)
    {Wert++;} // muss > 0 sein
}

return Wert; // Arg1 == Arg2 so immer 0
}

```

liefert nichts

Hinweis für Feldelement mit undefiniertem Inhalt und Null-Zeiger:

```

if (Wert == null)           // true
var Wert;                  // Variable deklariert aber ohne Wert
if (Wert == undefined)     // true
if (Wert == null)          // true
var Wert=10;               // Variable deklariert aber mitWert
if (Wert == undefined)     // false
if (Wert == null)          // false

```

Beispiel:

```

var Feld1 = new Array("4","3","2","1","0");
Feld1.sort();           // Feld1 ist ungültig
alert(Feld1.join());    // "0,1,2,3,4"

```

Beispiel für numerische Feldelemente:

```

function Sortiere(Arg1, Arg2)
{return Arg1 - Arg2;}      // Arg1 > Arg2 so positiver Wert
                          // Arg1 < Arg2 so negativer Wert
                          // Arg1 == Arg2 so 0
zeiger_auf_feld.sort(Sortiere);

```

.splice()

Feldelemente-Folge aus dem Feld entfernen ab Indexposition
optional neue Objekte in das Feld einfügen ab Indexposition
alle entfernte Feldelemente in einem neuen Feld liefern in der Reihenfolge des Entfernen
Felder haben die Objektklasse Script-Objekt Array
Syntax:

```
[ var Zeiger = ] zeiger_auf_feld.splice(Wert1, Wert2 [, werte_liste])
```

werte_liste	<p>optional Liste der einzufügenden Objekte ab Index laut Wert1 kommagetrennte Werte Anzahl der Listenelemente beliebig Wert kann z.B. sein Literal Referenz z.B. auf Feld wenn Feld, so dessen Elemente einzeln eingefügt aus Ausdruck aus Funktionsaufruf</p>
Wert1	<p>Index, ab 0 und < Länge des Feldes laut .length Startposition der zu entfernenden Feldelemente-Folge in der Anzahl laut Wert2 Startposition der einzufügenden Objekte laut werte_liste in der Anzahl laut Anzahl der Listenelemente</p>
Wert2	<p>Anzahl der Elemente, die entfernt werden sollen ab Index laut Wert1 <= Länge des Feldes laut .length > 0 wenn keine werte_liste kodiert, so wird nur entfernt 0 wenn keine werte_liste kodiert, so wird weder entfernt noch eingefügt</p>
Zeiger	<p>auf die neue Instanz, die automatisch erzeugt wird und alle entfernten Feldelemente in der Reihenfolge des Entfernen enthält, falls Wert2 > 0 ist</p>



Beispiel:

```
var Feld1 = new Array(0,0,1,2,3,4);
var Feld2 = Feld1.splice(0,1);
alert(Feld2.join()); // "0,1,2,3,4"
var Feld3 = Feld1.splice(0,2,-1,0);
alert(Feld3.join()); // "-1,0,1,2,3,4"
```

`.toString()` Feldinhalt als String liefern, wobei Feldelemente im String mit Komma getrennt sind
Syntax:

```
[ var Kette = ] zeiger_auf_feld.toString()
```

4.2.2.1.1. Array JScript-Objekt im Internet Explorer ab Version 5.5

ab IE 5.5 wurde das Array Script-Objekt um folgende Methoden erweitert:

`.pop()` letztes Feldelement entfernen und dann als Variable liefern
es wird Eigenschaft `.length` verändert
Feld hat die Objektklasse JScript-Objekt
Syntax:

```
[ var Zeiger = ] zeiger_auf_feld.pop()
```

Beispiel:

```
var names = new Array("Tom", "Mark", "Bart", "John");
names.push("Jim", "Richard", "Tim");
alert(names);
var last = names.pop();
alert("Last = " + last + " Other = " + names);
```

Beispiel:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.pop()); // "4"
alert(Feld.join()); // "0,1,2,3"
```

Beispiel für pop vor dem IE 5.5:

```
function pop()
{
    var lastElement = this[this.length - 1];
    this.length--;
    return lastElement;
}
```

```
var names = new Array("Tom", "Mark", "Bart", "John"); // Es wird die Eigenschaft .prototype erzeugt
```

```
// Prototyping der Instanz names
names.prototype.pop = pop;
```

```
// Auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.pop = pop;
```

`.push()` Feld durch Anhängen von neuen Elementen erweitern und danach die neue Feldlänge liefern
es wird Eigenschaft `.length` verändert
Feld hat die Objektklasse JScript-Objekt
Syntax:

```
[ var Wert = ] zeiger_auf_feld.push(Zeigerliste)
```

Zeigerliste kommagetrennte Zeiger auf anzuhängende Elemente
Anhängen laut Listenelemente-Folge

Beispiel:

```
var names = new Array("Tom", "Mark", "Bart", "John");
names.push("Jim", "Richard", "Tim");
alert(names);
var last = names.pop();
alert("Last = " + last + " Other = " + names);
vor IE 5.5 muss push und pop durch den Programmierer per Prototyping implementiert werden
```

Beispiel:

```
var Feld1 = new Array(0,1);
var Feld2_Quelle = new Array(2,3);
var Wert_Quelle = 4;
alert(Feld1.push(Wert_Quelle, Feld2_Quelle)); // 4 und nicht 5 !
alert(Feld1.join()); // "0,1,2,3" // 4 wird nicht angehängen
```



Beispiel für push vor dem IE 5.5:

```
function push()
{
    var sub = this.length;
    for (var i = 0; i < push.arguments.length; ++i)
    {
        this[sub] = push.arguments[i];
        sub++;
    }
}
```

```
var names = new Array("Tom", "Mark", "Bart", "John");
erzeugt
```

// Es wird die Eigenschaft .prototype

```
// Prototyping der Instanz names
names.prototype.push = push;
```

```
// auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.push = push;
```

.shift() erstes Feldelement entfernen und dann als Variable liefern
 es wird Eigenschaft .length verändert
 Feld hat die Objektklasse JScript-Objekt
 Syntax:
 [var Zeiger =] zeiger_auf_feld.shift()

Zeiger auf entferntes Feldelement

Beispiel:

```
var line = new Array("aaa", "bbb", "ccc", "ddd", "eee");
alert("first = " + line.shift() + " Other = " + line);
```

Beispiel:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.shift());      // 0
alert(Feld.join());      // "1,2,3,4"
```

.unshift() neue Feldelemente am Feldanfang einfügen und dann neue Länge des Array liefern
 Feld hat die Objektklasse JScript-Objekt
 Syntax:
 [var Wert =] zeiger_auf_feld.unshift(Zeigerliste)

Beispiel:

```
var line = new Array("ccc", "ddd", "eee");
line.unshift("aaa", "bbb");
alert(line);
```

Beispiel:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.unshift(-1));
alert(Feld.join());      // "-1,0,1,2,3,4"
```

Beispiel für unshift-Methode bei Browsern vor dem IE5.5:

```
function unshift()
{
    var i = unshift.arguments.length;
    for (var j = this.length - 1; j >= 0; --j)
    {this[j + i] = this[j]; }

    for (j = 0; j < i; ++j)
    {this[j] = unshift.argument[j]; }
}
```

```
Array.prototype.unshift = unshift;
```

Zur Konvertierung eines VisualBasic-Array in ein JScript-Array dient die Methode .toArray() eines VisualBasic-Array

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");
```



```

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchlüssel, Date);

    // und Daten-Referenz bilden
    var DatenReferenz = DatenSpeicher.Items();

    // und Feld bilden
    // VisualBasic-Feld erzeugen
    var DatenOhneSchlüssel_Feld = new VBArray(DatenReferenz);
    // und nach JScript-Feld konvertieren
    DatenOhneSchlüssel_Feld = DatenOhneSchlüssel_Feld.toArray();

    // und Daten anzeigen
    var DatenOhneSchlüssel_FeldLaenge = DatenOhneSchlüssel_Feld.length

    if (DatenOhneSchlüssel_FeldLaenge > 0)
    {
        for (var i = 0 ; i < DatenOhneSchlüssel_FeldLaenge; i++)
        {alert("Date " + i + " = " + DatenOhneSchlüssel_Feld [i]);}
    }
    else
    {alert("Dictionary ist leer!");}
}

```

4.2.2.1.2. Array Script-Objekt im Netscape ab Version 6.x (ab Javascript 1.5)

ab NS 6.x wurde das Array Script-Objekt um folgende Methoden erweitert:

.pop() letztes Feldelement entfernen und dann als Variable liefern
es wird Eigenschaft `length` verändert
Feld hat die Objektklasse Script-Objekt
Syntax: [var Zeiger =] zeiger_auf_feld.pop()

Beispiel:

```

var names = new Array("Tom", "Mark", "Bart", "John");
names.push("Jim", "Richard", "Tim");
alert(names);
var last = names.pop();
alert("Last = " + last + " Other = " + names);

```

Beispiel:

```

var Feld = new Array(0,1,2,3,4);
alert (Feld.pop());        // "4"
alert(Feld.join());        // "0,1,2,3"

```

Beispiel für pop vor dem NS 6.x:

```

function pop()
{
    var lastElement = this[this.length - 1];
    this.length--;
    return lastElement;
}

```

```

var names = new Array("Tom", "Mark", "Bart", "John");                      // Es wird die Eigenschaft .prototype erzeugt

```

```

// Prototyping der Instanz names
names.prototype.pop = pop;

```

```

// Auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.pop = pop;

```



.push() Feld durch Anhängen von neuen Elementen erweitern und danach die neue Feldlänge liefern
 es wird Eigenschaft `.length` verändert
 Feld hat die Objektklasse `Script-Objekt`
 Syntax:

```
[ var Wert = ] zeiger_auf_feld.push(Zeigerliste)
```

Zeigerliste kommagetrennte Zeiger auf anzuhängende Elemente
 Anhängen laut Listenelemente-Folge

Beispiel:

```
var names = new Array("Tom", "Mark", "Bart", "John");
names.push("Jim", "Richard", "Tim");
alert(names);
var last = names.pop();
alert("Last = " + last + " Other = " + names);
vor IE 5.5 muss push und pop durch den Programmierer per Prototyping implementiert werden
```

Beispiel:

```
var Feld1                      = new Array(0,1);
var Feld2_Quelle               = new Array(2,3);
var Wert_Quelle                = 4;
alert(Feld1.push(Wert_Quelle , Feld2_Quelle));        // 4 und nicht 5 !
alert(Feld1.join());           // "0,1,2,3"            // 4 wird nicht angehängen
```

Beispiel für push vor dem NS 6.x:

```
function push()
{
    var sub = this.length;
    for (var i = 0; i < push.arguments.length; ++i)
    {
        this[sub] = push.arguments[i];
        sub++;
    }
}

var names = new Array("Tom", "Mark", "Bart", "John");        // Es wird die Eigenschaft .prototype
erzeugt

// Prototyping der Instanz names
names.prototype.push = push;

// auch möglich: Prototyping des Script-Objektes Array generell
Array.prototype.push = push;
```

.shift() erstes Feldelement entfernen und dann als Variable liefern
 es wird Eigenschaft `.length` verändert
 Feld hat die Objektklasse `Script-Objekt`
 Syntax:

```
[ var Zeiger = ] zeiger_auf_feld.shift()
```

Zeiger auf entferntes Feldelement

Beispiel:

```
var line = new Array("aaa", "bbb", "ccc", "ddd", "eee");
alert("first = " + line.shift() + " Other = " + line);
```

Beispiel:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.shift());        // 0
alert(Feld.join());         // "1,2,3,4"
```

.unshift() neue Feldelemente am Feldanfang einfügen und dann neue Länge des Array liefern
 Feld hat die Objektklasse `Script-Objekt`
 Syntax:

```
[ var Wert = ] zeiger_auf_feld.unshift(Zeigerliste)
```

Beispiel:

```
var line = new Array("ccc", "ddd", "eee");
line.unshift("aaa", "bbb");
alert(line);
```

Beispiel:



```
var Feld = new Array(0,1,2,3,4);
alert (Feld.unshift(-1));
alert(Feld.join());    // "-1,0,1,2,3,4"
```

Beispiel für unshift-Methode bei Browsern vor dem NS 6.x:

```
function unshift()
{
    var i = unshift.arguments.length;
    for (var j = this.length - 1; j >= 0; --j)
    {this[j + i] = this[j]; }

    for (j = 0; j < i; ++j)
    {this[j] = unshift.argument[j]; }
}
```

Array.prototype.unshift = unshift;

4.2.2.2. **Array Script-Objekt aus Literalen**

Diese Variante eines Feldes ist eine vereinfachte Kodierung für Elemente vom Literaltyp (String- oder numerisches Literal). Mit der Instanziierung wird automatisch die new-Anweisung mit internem Konstruktor aufgerufen. Mit diesem Feld findet auch die Anweisung for .. in Anwendung.

Syntax:

```
var Zeiger = {literal_liste}
```

literal_liste: kommagetrennte Elemente

Element

mit Aufbau "kette1":"kette2"

Doppelpunkt muss kodiert werden

kette 1 Eigenschaft des Literal-Objektes
nur Plain-Text

kette2 Wert der Eigenschaft des Literal-Objektes
nur Plain-Text

Beispiel:

```
var LiteralObjekt = {"Vorname":"Otto", "Nachname":"Waalkes"};
```

```
var Index = "";
var Menge = {"a" : "Athen", "b" : "Berlin", "c" : "Paris", "d" : "Kairo"};
var Kette = "";
```

SchleifenAnweisung : // ein Label (Marke)

```
{
    for (Index in Menge)
    {
        Kette = "Hauptstadt von ";

        switch (Menge[Index])
        {
            case "Berlin":                      {
                Kette += "Deutschland: " + Menge[Index];
                // nicht weiter auf Athen und Kairo prüfen
                break;
            }

            case "Athen":                      {
                Kette += "Griechenland: " + Menge[Index];
                // kein break, also noch auf Kairo prüfen
            }

            case "Kairo":                      {
                Kette += "Ägypten: " + Menge[Index]; }

            default:                            {
                // im Falle von Paris:
                //                      Kairo wird nie angezeigt, da im Feld
                //                      hinter Paris
            }
        }
    }
}
```



```

        }
        break SchleifenAnweisung;
    }
    {alert(Kette);} // nicht bei Paris abgearbeitet
}

```

Eigenschaften:

wie Array als Script-Objekt

Methoden:

wie Array als Script-Objekt inklusive der Methoden ab IE 5.5 bzw. 6.x

4.2.3. Boolean Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.

siehe auch Basis-Datentyp boolean

Erzeugung:

```
[ var Zeiger2 = ] new Boolean([Zeiger1]);
```

```
var Zeiger2 = Zeiger1;
```

Zeiger 1

ausdruck	Ausdruck muss true oder false liefern und niemals void(ausdruck)
Wert	true oder false oder beliebiger Wert wenn 0, -0, +0, null, false, NaN, undefined oder Leerkette "" so wird false verwendet sonst wird true verwendet z.B. "false" wird als true verwendet Nicht-Null-Zeiger wird als true verwendet 1 wird als true verwendet
Referenz	auf Objekt mit Wert (siehe oben)
Default	Boolean-Wert false

Hinweis: `zeiger_auf_boolean_objekt.toString()` liefert String-Werte "true" bzw. "false"
`zeiger_auf_boolean_objekt.valueOf()` liefert Boolean-Werte true bzw. false

logischen Vergleich mit vor bzw. nach expliziter Wertzuweisung:

Beispiel:

```

var InitWert = false;
var x = new Boolean(InitWert);
if (x) // falscher Vergleich da immer true, egal welcher Initwert
if (x == false) // korrekter Vergleich der true liefert

// neuer Wert
x = false;
if (x == false) // true
if (!x) // true

```

Konvertierung in Boolean-Wert:

Booleanwerte sind true (nicht "true" etc.)
false (nicht "false" etc.)

Syntax: `[var Zeiger2 =] Boolean(Zeiger1);`

Zeiger 1

ausdruck	Ausdruck muss true oder false liefern und niemals void(ausdruck)
Wert	true oder false oder beliebiger Wert wenn 0, -0, +0, null, false, NaN, undefined oder Leerkette "" so wird false verwendet sonst wird true verwendet z.B. "false" wird als true verwendet Nicht-Null-Zeiger wird als true verwendet 1 wird als true verwendet
Referenz	auf Objekt mit Wert (siehe oben)
Default	Boolean-Wert false

Eigenschaften:

keine

Methoden:

keine

4.2.4. Date Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.

Dieses Objekt verwaltet Datum und Zeit.

Berechnungen mit Date Script-Objekt sind nur auf Basis von Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit möglich (positiver bzw. negativer Wert der Anzahl der Millisekunden).

Zeitzone-Berechnungen sind nur per .getTimezoneOffset() möglich.

maximales Datum muss im Jahr 1970 + 285,616 bzw. - 285,616 Jahre liegen
+ 100000000 bzw. -100000000 Tage liegen

Weltzeit: Universal Coordinated Time (UTC)

entspricht GMT (Greenwich Mean Time)

siehe auch Basis-Datenstruktur date

Erzeugung:

var Zeiger = new Date();

Objekt wird initialisiert mit der positiver Anzahl der Millisekunden

ab dem 01.01.1970 0 Uhr Weltzeit (UTC)**bis zum aktuellen Zeitpunkt der Instanziierung des Objektes in Weltzeit**

var Zeiger = new Date(Wert);

Wert zum Initialisieren des Objektes auf **Weltzeit**Anzahl der Millisekunden relativ zu dem 01.01.1970 0 Uhr **Weltzeit**

positive, negativ

var Zeiger = new Date(Jahr, Monat, Tag [, Stunden [, Minuten [, Sekunden [, Millisekunden]]]]);

alle Argumente dienen zum Initialisieren des Objektes auf **Weltzeit**

Lücke in der Argumentenliste nicht zulässig:

0 kodieren wenn Argument nicht verwendet werden soll

bei Argumenten, deren Werte über den gültigen Wertebereich hinausgehen, ist die Scriptmaschine **fehlertolerant**:

Beispiel: 61 als Angabe für Sekunden --> Script verwendet 1 Minute und 1 Sekunde und berechnet alle darüberliegende Werte neu, falls diese ebenfalls Überschreitungen des Wertebereiches liefern

Ursache: Scriptmaschine rechnet **immer** in Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit (automatische Umwandlung der Werte in Millisekunden)

Jahr	Integer	
	immer vierstellig	z.B. 1976
	minimal 1970	
Monat	String mit englischem Monatsnamen	
	oder Integer	0 bis 11
		0 entspricht "January"
		11 entspricht "December"
Tag	Integer	
	1 bis 31	
	Achtung: Der Programmierer muss Schaltjahre und Anzahl der Tage in dem Monaten wissen.	
Stunden	Integer	
	0 bis 23	
	24 Stunden sind 1 Tag	
	Default ist 0	
Minuten	Integer	
	0 bis 59	
	60 Minuten sind 1 Stunde	
	Default ist 0	
Sekunden	Integer	
	0 bis 59	



60 Sekunden sind 1 Minute
Default ist 0

Millisekunde Integer
0 bis 999
1000 Millisekunden sind 1 Sekunde
Default ist 0

Beispiel 1:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookieen_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
        && ( Index < AnzahlCookies)
    );

    return CookieWert;
}
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
```



```

<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachennamen definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitpunktUTC = Zeitpunkt.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitpunktUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()

```



```

{
    var AnzahlMillisekundenProTag = 86400000;

    var DatumDokumentErzeugung = new Date(document.fileCreatedDate);

    var DatumHeute = new Date();

    var TagesDifferenz = ( DatumHeute.getTime()
                          - DatumDokumentErzeugung.getTime()
                          )
                        / AnzahlMillisekundenProTag;

    alert( "Dokument erzeugt am " + DatumDokumentErzeugung
          + "\n.... also vor " + parseInt(TagesDifferenz) + " Tagen"
          );
}
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

Beispiel 5 für Datumsanzeige in deutscher Norm:

```

HTML>
<HEAD>
<SCRIPT LANGUAGE=JavaScript1.2>
<!--
        //          Datumsanzeige in deutscher Norm

//*****
//
//          Vom Programmierer ist nichts zu verändern
//
//*****

// Felder der deutschen Datumsangaben
var WochenTag=new Array(
    "Sonntag",
    "Montag",
    "Dienstag",
    "Mittwoch",
    "Donnerstag",
    "Freitag",
    "Sonnabend"
);

var Monate=new Array(
    "Januar",
    "Februar",
    "März",
    "April",
    "Mai",
    "Juni",
    "Juli",
    "August",
    "September",
    "Oktober",
    "November",
    "Dezember"
);

// Daten des aktuellen Datums ermitteln
var Datum_Aktuell=new Date();
var Datum_Aktuell_Jahr=Datum_Aktuell.getYear();
var Datum_Aktuell_WochenTag=Datum_Aktuell.getDay();
var Datum_Aktuell_Monat=Datum_Aktuell.getMonth();
var Datum_Aktuell_Tag=Datum_Aktuell.getDate();

// Anpassungen der Daten des aktuellen Datums
// wenn Jahresangabe < 1000 so 1900 dazuaddieren

```



```

if (Datum_Aktuell_Jahr < 1000) {Datum_Aktuell_Jahr+=1900;}

//      Vornull einbasteln
if (Datum_Aktuell_Tag<10)   {Datum_Aktuell_Tag="0"+Datum_Aktuell_Tag;}

// Formatierte Anzeige des aktuellen Datums in deutscher Norm
document.write(
    WochenTag[Datum_Aktuell_WochenTag]
    + ", "
    + Datum_Aktuell_Tag
    + "."
    + Monate[Datum_Aktuell_Monat]
    + " "
    + Datum_Aktuell_Jahr
);

//-->
</SCRIPT>
<BODY>
</BODY>
</HTML>

```

Zugriff:

date_instanz.eigenschaft
date_instanz.methode()

Eigenschaften:

keine

Methoden:

wird eine nachfolgend beschriebene Methode als deprecated genannt, so gilt das nur für den IE.

Empfehlung: Methode, die deprecated ist, auch im NS nicht verwenden.

.getDate()	Tag des Monats in lokaler Zeit liefern
Syntax:	
	[Wert =] object.getDate()
	objekt Zeiger auf Instanz von Script-Objekt Date
	Wert Integer
	1 bis 31
.getDay()	Tag der Woche in lokaler Zeit liefern
Syntax:	
	[Wert =] object.getDay()
	objekt Zeiger auf Instanz von Script-Objekt Date
	Wert Integer
	0 bis 6
	0 ist Sonntag
	6 ist Samstag
.getFullYear()	Jahreszahl vierstellig in lokaler Zeit liefern
Syntax:	
	[Wert =] object.getFullYear()
	objekt Zeiger auf Instanz von Script-Objekt Date
	Wert Integer
	maximal 1970 + bzw. -285,616 Jahre
.getHours()	Stunden in lokaler Zeit liefern
Syntax:	
	[Wert =] object.getHours()
	objekt Zeiger auf Instanz von Script-Objekt Date
	Wert Integer
	0 bis 23
.getMilliseconds()	Millisekunden in lokaler Zeit liefern
Syntax:	
	[Wert =] object.getMilliseconds()



	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 999
.getMinutes()	Minuten in lokaler Zeit liefern Syntax: [Wert =] object.getMinutes()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 59
.getMonth()	Monat in lokaler Zeit liefern Syntax: [Wert =] object.getMonth()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 11
.getSeconds()	Sekunden in lokaler Zeit liefern Syntax: [Wert =] object.getSeconds()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 59
.getTime()	Zeitwert als Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit liefern positiv oder negativ möglich (wenn < 0 so vor obigen Datum) Syntax: [Wert =] object.getTime()	
	Wert	Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit wenn < 0 so vor 01.01. 1970 0 Uhr Weltzeit
.getTimezoneOffset()	Minuten der lokalen Zeit als Differenz zur Weltzeit liefern, also das Offset der lokalen Zeitzone Syntax: [Wert =] object.getTimezoneOffset()	
	Wert	Integer ab 0
.getUTCDate()	Tag des Monats in Weltzeit liefern Syntax: [Wert =] object.getUTCDate()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 1 bis 31
.getUTCDay()	Tag der Woche in Weltzeit liefern Syntax: [Wert =] object.getUTCDay()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 6 0 ist Sonntag 6 ist Samstag
.getUTCFullYear()	Jahreszahl vierstellig in Weltzeit liefern Syntax: [Wert =] object.getUTCFullYear()	



	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer maximal 1970 + bzw. -285,616 Jahre
.getUTCHours()	Stunden in Weltzeit liefern Syntax: [Wert =] object.getUTCHours()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 23
.getUTCMilliseconds()	Millisekunden in Weltzeit liefern Syntax: [Wert =] object.getUTCMilliseconds()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 999
.getUTCMinutes()	Minuten in Weltzeit liefern Syntax: [Wert =] object.getUTCMinutes()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 59
.getUTCMonth()	Monat in Weltzeit liefern Syntax: [Wert =] object.getUTCMonth()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 11
.getUTCSeconds()	Sekunden in Weltzeit liefern Syntax: [Wert =] object.getUTCSeconds()	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 59
.getYear()	deprecated	
.parse()	Datum als String parsen und Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit liefern Datum als String in den Basis-Datentyp date konvertieren Syntax: [Wert =] object.parse(Kette)	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Kette	String mit Aufbau n monat tag, jiji hh:mm:ss x z n monat/tag/jiji hh:mm:ss x z n monat-tag-jiji hh:mm:ss x z alle Elemente sind optional, wobei mindestens 1 Element kodiert werden muss n nur englischer Wochentagname wenn kalendarisch falscher Wochentag, so



automatisch korrigiert

monat	nur englischer Name
tag	1 bis 31, ohne Vornull
jjjj	für Jahr 2 oder vierstellig wenn 2 stellig so unter 2000 gemeint (ab 1970)
hh und mm und ss	wie üblich für Stunde Minute und Sekunde
xx	AM oder PM bei 12-Stunden-Uhr wenn hh bis ss nicht passend, so Fehler geliefert z.B. 23:00 PM gibt es nicht, also Fehler erzeugt
z	Zeitzone zB. UTC oder GMT

Wert Integer
Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit
wenn n kodiert wurde und n kalendarisch ein falscher Wochentag ist, so
wird n automatisch korrigiert, bevor Wert geliefert wird

Beispiele:

"Jan 5, 1996 08:47:00"
 "November 1, 1997 10:15:00 "
 "November 1, 1997 10:15 AM"
 "7/20/96 08:47:00"
 "7-20-96 08:47:00"
 "Tuesday November 9 1990"
 "7-20-96 08:" es muss Doppelpunkt kodiert sein, damit 08 als Stundenangabe interpretiert wird
 "7-20-96 08:47 GMT"

.setDate() Tag des Monats in lokaler Zeit setzen
 wenn Tag > Anzahl Tage im Monat, so erfolgt automatisch Monatswechsel und Korrektur aller
 Angaben (inklusive Jahr)

Syntax:

object.setDate(Wert)

objekt Zeiger auf Instanz von Script-Objekt Date

Wert Integer
1 bis 31

.setFullYear() Jahreszahl vierstellig in lokaler Zeit setzen und optional Monat wie Tag im Monat
 sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
 Syntax:

object.setFullYear(Wert1 [,Wert2 [,Wert3]])

objekt Zeiger auf Instanz von Script-Objekt Date

Wert1 Jahr
Integer
maximal **1970 + bzw. -285,616 Jahre**

Wert2 Monat
Integer
0 bis 11
Default laut .getMonth()

Wert3 Tag im Monat
Integer
1 bis 31
Default laut .getDate()

.setHours() Stunden in lokaler Zeit setzen und optional Minuten, Sekunden, Millisekunden
 sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
 Syntax:

object.setHours(Wert1 [,Wert2 [,Wert3 [,Wert4]])]

objekt Zeiger auf Instanz von Script-Objekt Date



Wert1 Stunden
Integer
0-23

Wert2 Minuten
Integer
0 bis 59
Default laut .getMinutes()

Wert3 Sekunden
Integer
0 bis 59
Default laut .getSeconds()

Wert4 Millisekunden
Integer
0 bis 999
Default laut .getMilliseconds()

.setMilliseconds()

Millisekunden in lokaler Zeit setzen

Syntax:

object.setMilliseconds(Wert)

objekt Zeiger auf Instanz von Script-Objekt Date

Wert Integer
0 bis 999**.setMinutes()**

Minuten in lokaler Zeit setzen und optional Sekunden, Millisekunden

sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert

Syntax:

object.setMinutes(Wert1 [,Wert2 [,Wert3]])

objekt Zeiger auf Instanz von Script-Objekt Date

Wert1 Minuten
Integer
0 bis 59Wert2 Sekunden
Integer
0 bis 59
Default laut .getSeconds()Wert3 Millisekunden
Integer
0 bis 999
Default laut .getMilliseconds()**.setMonth()**

Monat in lokaler Zeit setzen und optional Tag im Monat

sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert

Syntax:

object.setMonth(Wert1 [,Wert2])

objekt Zeiger auf Instanz von Script-Objekt Date

Wert1 Monat
Integer
0 bis 11Wert2 Tag im Monat
Integer
1 bis 31
Default laut .getDate()**.setSeconds()**

Sekunden in lokaler Zeit setzen und optional Millisekunden

sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert

Syntax:

object.setSeconds(Wert1 [,Wert2])



	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Sekunden Integer 0 bis 59
	Wert2	Millisekunden Integer 0 bis 999 Default laut .getMilliseconds()
.setTime()	Zeitwert in Weltzeit setzen, auch vor 1970 Syntax: object.getTime(Wert)	
	Wert	Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit wenn < 0 so vor obigem Datum
.setUTCDate()	Tag des Monats in Weltzeit setzen wenn Tag > Anzahl Tage im Monat, so erfolgt automatisch Monatswechsel und Korrektur aller Angaben (inklusive Jahr) Syntax: object.setUTCDate(Wert)	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 1 bis 31
.setUTCFullYear()	Jahreszahl vierstellig in Weltzeit setzen und optional Monat wie Tag im Monat sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Syntax: object.setUTCFullYear(Wert1 [,Wert2 [,Wert3]])	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Jahr Integer maximal 1970 + bzw. -285,616 Jahre
	Wert2	Monat Integer 0 bis 11 Default laut .getMonth()
	Wert3	Tag im Monat Integer 1 bis 31 Default laut .getDate()
.setUTCHours()	Stunden in Weltzeit setzen und optional Minuten, Sekunden, Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Syntax: object.setUTCHours(Wert1 [,Wert2 [,Wert3 [,Wert4]])	
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Stunden Integer 0-23
	Wert2	Minuten Integer 0 bis 59 Default laut .getMinutes()
	Wert3	Sekunden Integer 0 bis 59 Default laut .getSeconds()



	Wert4	Millisekunden Integer 0 bis 999 Default laut .getMilliseconds()
.setUTCMilliseconds()		Millisekunden in Weltzeit setzen Syntax: object.setUTCMilliseconds(Wert)
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert	Integer 0 bis 999
.setUTCMinutes()		Minuten in Weltzeit setzen und optional Sekunden, Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Syntax: object.setUTCMinutes(Wert1 [,Wert2 [,Wert3]])
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Minuten Integer 0 bis 59
	Wert2	Sekunden Integer 0 bis 59 Default laut .getSeconds()
	Wert3	Millisekunden Integer 0 bis 999 Default laut .getMilliseconds()
.setUTCMonth()		Monat in Weltzeit setzen und optional Tag im Monat sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Syntax: object.setUTCMonth(Wert1 [,Wert2])
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Monat Integer 0 bis 11
	Wert2	Tag im Monat Integer 1 bis 31 Default laut .getDate()
.setUTCSeconds()		Sekunden in Weltzeit setzen und optional Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Syntax: object.setUTCSeconds(Wert1 [,Wert2])
	objekt	Zeiger auf Instanz von Script-Objekt Date
	Wert1	Sekunden Integer 0 bis 59
	Wert2	Millisekunden Integer 0 bis 999 Default laut .getMilliseconds()
.setYear()		deprecated
.toString()		kompletten Inhalt des Date-Objektes als String liefern



Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.toGMTString()

deprecated

.toLocaleString()

kompletten Inhalt des Date-Objektes als String in lokaler Einstellung auf User-PC liefern
nur für Jahre ab 2000

aber für Jahre von 1601 bis 1999 immer laut lokale Einstellungen des Betriebssystems auf User-PC
siehe .toLocaleDateString()

Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toLocaleString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.toLocaleDateString()

kompletten Inhalt des Date-Objektes als String in lokaler Einstellung des Betriebssystems auf User-PC
liefern

immer verwenden für Jahre von 1601 bis 1999 --> siehe .toLocaleString()

Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toLocaleDateString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.toLocaleTimeString()

Zeitwert des Date-Objektes als String in lokaler Einstellung des Betriebssystems auf User-PC
liefern

Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toLocaleTimeString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.toString()

Datum als String geliefert (je nach Ländereinstellung des Windows)

Beispiel: Thu Aug 18 04:37:43 GMT-0700 (Pacific Daylight Time) 1983

Syntax:

```
[ var Kette = ] zeiger_auf_date_objekt.toString()
```

.toTimeString()

Zeitwert des Date-Objektes als String liefern

Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toTimeString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.toUTCString()

Zeitwert des Date-Objektes als String in Weltzeit liefern

Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !

Syntax:

```
[ var Kette = ] object.toUTCString()
```

objekt Zeiger auf Instanz von Script-Objekt Date

Kette String

.UTC()

Inhalt des Date-Objektes setzen nach Weltzeit

(auch wenn Objekt bereits initialisiert wurde per new Anweisung)

sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert

Syntax:



```
[ var Wert8 = ] object.UTC(Wert1 ,Wert2 ,Wert3 [, Wert4 [, Wert5 [,Wert6 [,Wert7]]]])
```

objekt	Zeiger auf Instanz von Script-Objekt Date
Wert1	Jahr Integer maximal 1970 + bzw. -285,616 Jahre
Wert2	Monat Integer 0 bis 11
Wert3	Tag im Monat Integer 1 bis 31
Wert4	Stunden Integer 0-23 Default laut .getHours()
Wert5	Minuten Integer 0 bis 59 Default laut .getMinutes()
Wert6	Sekunden Integer 0 bis 59 Default laut .getSeconds()
Wert7	Millisekunden Integer 0 bis 999 Default laut .getMilliseconds()
Wert8	Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit wenn < 0 so vor obigem Datum

.valueOf() Wert als Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr Weltzeit (UTC)
wenn Datum vor dem 1.1.1970 0 Uhr Weltzeit so Wert negativ
Syntax

```
[ var Wert = ] zeiger_auf_date_objekt.valueOf()
```

4.2.4. Enumerator JScript-Objekt des Internet Explorer

Dieses Objekt ist eine Komponente der Scriptsprache.
dient der erweiterten Verwaltung von von Elementen einer Collection (nicht reines Array)
durch Verwendung eines internen Index für den Zugriff auf ein Element der Collection
ist Analogon zu dem Array Script-Objekt für reine Felder

Erzeugung:

```
var Zeiger2 = new Enumerator([Zeiger1])
```

Zeiger1 auf eine Collection

Zeiger2 auf Enumerator-Objekt

Beispiel Laufwerke auf dem PC des Users ermitteln:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd(); DateiSystem_Laufwerke.moveNext() )
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
```



```

    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

Eigenschaften:

keine

Methoden:

```

.atEnd()           prüfen    auf Erreichen des Ende der Collection
                   auf Zustand undefined eines Elementes der Collection
                   auf leere Collection
                   verwenden mit .moveNext() innerhalb einer Schleife

```

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd(); DateiSystem_Laufwerke.moveNext())
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {

```



```

        // ist bereit, dann Volume-Bezeichner ermittelbar
        LaufwerkName = Laufwerk.VolumeName;
    }
    else
    { LaufwerkName = "[Drive not ready]";}
}

Kette += LaufwerkName + "\n";
}
alert (Kette);

```

.item() Element einer Collection liefern laut aktueller Position in der Collection zur Positionierung innerhalb der Collection wird ein interner Index verwendet

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem                = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke     = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd();DateiSystem_Laufwerke.moveNext())
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]";}
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

.moveFirst() aktuelle Position innerhalb der Collection auf das erste Element der Collection setzen (auch wenn Collection leer ist) zur Positionierung innerhalb der Collection wird ein interner Index verwendet Element an aktueller Position ermitteln per .item()

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem                = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke     = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;

```



```

var LaufwerkName; // Netzname oder Volume-Bezeichner

// erstes Laufwerk im Dateisystem einstellen (erstes Element im Enumerator-Objekt)
DateiSystem_Laufwerke.moveFirst();
do
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";

    // nächstes Laufwerk positionieren (nächstes Element im Enumerator-Objekt)
    DateiSystem_Laufwerke.moveNext();
}
while (!DateiSystem_Laufwerke.atEnd());

alert(Kette);

```

.moveNext() aktuelle Position innerhalb der Collection auf das nächste Element der Collection setzen (auch wenn Collection leer ist)
zur Positionierung innerhalb der Collection wird ein interner Index verwendet
Element an aktueller Position ermitteln per .item()

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem                      = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke           = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

// erstes Laufwerk im Dateisystem einstellen (erstes Element im Enumerator-Objekt)
DateiSystem_Laufwerke.moveFirst();
do
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

```



```

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";

    // nächstes Laufwerk positionieren (nächstes Element im Enumerator-Objekt)
    DateiSystem_Laufwerke.moveNext();
}
while (!DateiSystem_Laufwerke.atEnd());

alert(Kette);

```

4.2.5. error JScript-Objekt im Internet Explorer

Dieses Objekt

ist eine Komponente der Scriptsprache.
 enthält Fehlerinformationen zu einem Run-Time-Error
 wird verwendet für Erzeugung eines privaten Run-Time-Errors
 von Standard-Run-Time-Error:

Errornummer	32-Bit	Error-Code-Gruppe
	Bit 0 bis 15	Error-Code innerhalb der Gruppe

Erzeugung:

```
var Zeiger = new Error([ [Wert ,] Kette])
```

Wert	Fehlernummer Integer >= 0 Standard ist 0
Kette	Beschreibung des Fehlers Standard ist Leerkette

Beispiel für private Ausnahmebedingung ohne Error-Objekt

```

function ErzeugeAusnahme Bedingung(Ausnahme Bedingung)
{
    this.Ausnahme Bedingung=Ausnahme Bedingung;
    this.AusnahmeArt="UserException";
}

function HoleMonatAlsString (MonatsNummer) // liefert Monat als String
{
    var Index = MonatsNummer -1;

    var MonatsFeld =new Array("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

    if (MonatsFeld[Index] != null)
    {return MonatsFeld[Index];}
    else
    {
        var UserDefinierteAusnahmeBedingung =
            new ErzeugeAusnahme Bedingung ("FalscheMonatsNummer");
        throw UserDefinierteAusnahmeBedingung;
    }
}

```




```
}
```

Beispiel für private Ausnahmebedingung mit Error-Objekt

```
function HoleMonatAlsString (MonatsNummer) // liefert Monat als String
{
    var Index = MonatsNummer -1;

    var MonatsFeld =new Array("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

    if (MonatsFeld[Index] != null)
    {return MonatsFeld[Index];}
    else
    {
        var UserDefinierteAusnahmeBedingung = new Error(8888 , "test");

        throw UserDefinierteAusnahmeBedingung;
    }
}
```

Beispiele für internes Error-Objekt "e" bei Verwendung von try ... catch:

Beispiel 1:

```
function Anzeige(Kette)
{alert(Kette); }

try {Anzeige("try1");
{
    try
    {
        throw "Das ist eine Fehlerbedingung";
        Anzeige("try 2");
    }
    catch(e)
    {
        if (e == "Das ist eine Fehlerbedingung" )
        {Anzeige("catch2 " + e); }
    }
    finally { Anzeige("finally2"); }
}
catch(e) { Anzeige("catch1 " + e); }
finally { Anzeige("finally1");}
```

Beispiel 2:

```
function AlterErmittleIn(Wert)
{
    if(Wert < 0)
    {throw new Error(8888,"Fehler: Altersangabe muss >= 0 sein");}
}

try
{AlterErmittleIn (-5);} // aktiviert throw
catch(e)
{alert (e.message);}
```

Eigenschaften:

.description	Beschreibung des Run-Time-Errors deprecated: dafür Eigenschaft .message verwenden, die identische Funktion hat
.message	Beschreibung des Run-Time-Errors identisch mit Eigenschaft .description, die aber deprecated ist

Beispiel:

```
function AlterErmittleIn(Wert)
{
    if(Wert < 0)
    {throw new Error("Fehler: Altersangabe muss >= 0 sein");}
}

try
```



```
{AlterErmitteln (-5);} // aktiviert throw
catch(e)
{alert (e.message);}
```

.name Ausnahmetyp des Run-Time-Error liefern
privater Run-Time-Error

"Error"	Standard-Run-Time-Error
"ConversionError"	Konvertierungsfehler
"RangeError"	Bereichfehler z.B. Feldlänge negativ
"ReferenceError"	Referenz-Fehler z.B. Feld mit negativer Anzahl von Feldelementen soll erzeugt werden
"RegExpError"	Fehler bei regular Expression (RegExp-Objekt)
"SyntaxError"	
"TypeError"	Typfehler z.B. bei Wertzuweisung
"URIError"	falscher Uniform Resource Indicator (URI) z.B. bei encode()

Beispiel:

```
function AlterErmitteln(Wert)
{
    if(Wert < 0)
    {throw new Error("Fehler: Altersangabe muss >= 0 sein");}
}

try
{AlterErmitteln (-5);} // aktiviert throw
catch(e)
{alert (e.message + " " + e.name);}
```

.number Nummer des Run-Time-Error

Beispiel:

```
function AlterErmitteln(Wert)
{
    if(Wert < 0)
    {throw new Error(8888,"Fehler: Altersangabe muss >= 0 sein");}
}

try
{AlterErmitteln (-5);} // aktiviert throw
catch(e)
{alert (e.message + " " + e.number);}
```

Methoden:

keine

4.2.6. Function Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.

Erzeugung:

siehe Anweisung function

Syntax:

```
zeiger_auf_funktion.eigenschaft
zeiger_auf_funktion.methode()
```

zeiger_auf_funktion laut Erzeugung

Hinweis: zeiger_auf_funktion.toString() liefert den Quellcode

Eigenschaften:**.arguments**

Zeiger auf das Script-Objekt arguments
alle Eigenschaften des Script-Objektes arguments können referenziert werden
arguments nur verfügbar während der Ausführung der Funktion

Beispiel:

```
function Test1()
{
    var AnzahleArgumente = arguments.length;
    var Kette = "Anzahl Argumente = " + AnzahleArgumente + "\n";

    for ( var i = 0; i < AnzahleArgumente; i++)
    {Kette = "Argument[" + i + "]= " + arguments[i] + "\n"; }

    alert(Kette);
}
```

function Test2(Wert1,Wert2)



```
{
    var AnzahleArgumente = arguments.length;
    var Kette = "Anzahl Argumente = " + AnzahleArgumente + "\n";

    for ( var i = 0; i < AnzahleArgumente; i++)
    { Kette = "Argument[" + i + "] = " + arguments[i] + "\n"; }

    alert(Kette);
}

Test1();
Test2(10,20);
```

.caller	Zeiger auf den Aufrufer der Funktion Aufrufer ist eine andere Funktion (außer bei Rekursion)
---------	---

.length	Wert laut zeiger_auf_funktion.arguments.length
---------	--

Methoden:

<code>.apply()</code>	ein anderes Objekt anstelle des arguments Script-Objekt verwenden
<code>.call()</code>	eine Methode eines anderen Objektes aufrufen: Argumentenliste beachten

4.2.7. Math Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.
dient für mathematische Operationen

Erzeugung:

keine, da vom Browser implementiert

Zugriff:

```
[ var Wert = ] Math.eigenschaft
[ var Wert = ] Math.methode()
```

Wert ist NaN, wenn Operation nicht erfolgreich ist

Beispiel für Erzeugung einer Zufallszahl:

```
function ZufallsWertGanzzahligErzeugen1(MaxWert) // Zufallszahl von          inklusive 0
//                                          // bis inklusive MaxWert
//                                          // erzeugen als ganze Zahl
//                                          // liefert 0 wenn MaxWert 0 ist
// MaxWert:      wenn Gleitkomma, so vor Verwendung nach ganzzahlig konvertiert
//              wenn negativ, so Absolutwert verwendet und wenn dann Gleitkomma, dann zu ganzer Zahl
//              Bsp.:              -22.5 wird zu 22.5 wird zu 22
//              jedoch NICHT -22.5 wird zu -23  wird zu 23
{
    var ZufallsZahl=0;

    // zu positiv
    MaxWert=Math.abs(MaxWert);

    // zu ganzer Zahl
    MaxWert=Math.floor(MaxWert);

    if (MaxWert > 0)
    {
        ZufallsZahl=Math.random(); // ab eventuell einschliesslich 0 bis unter 1, Gleitkomma
// eventuell: falls nicht inklusive 0, so doch 0
// letztendlich erzeugbar durch nachfolgendes Runden

        if (ZufallsZahl > 0) // Bps. MaxWert 3 Zufallszahl  0,412345
// 0,612345
// 0,0015545
// 0,29456

        {
            // Kommaverschiebung
            while ((ZufallsZahl * 10) <= MaxWert)
            {ZufallsZahl = ZufallsZahl * 10;} // 0,412345  0,612345 1,5545  2,9456

            ZufallsZahl= Math.round(ZufallsZahl); // Kommastellen abschneiden durch runden
// ab inklusive 0,5 aufwärts, sonst abwärts
// 0 1 2 3

            // falls Runden einen Wert > MaxWert ergab, so den Wert begrenzen auf MaxWert
        }
    }
}
```



```

        if (ZufallsZahl > MaxWert)
        {ZufallsZahl = MaxWert;}
    }

    return ZufallsZahl;
}

function ZufallsWertGanzzahligErzeugen(MaxWert) // Zufallszahl von    inklusive 0
                                                //                    bis inklusive MaxWert
                                                //                    erzeugen als ganze Zahl
                                                // liefert 0 wenn MaxWert 0 ist
// MaxWert:    wenn Gleitkoma, so vor Verwendung nach ganzzahlig konvertiert
//              wenn negativ, so Absolutwert verwendet und wenn dann Gleitkoma, dann zu ganzer Zahl
//              Bsp.:    -22.5 wird zu 22.5 wird zu 22
//              jedoch NICHT -22.5 wird zu -23 wird zu 23
{
    var ZufallsZahl=0;

    // zu positiv
    MaxWert=Math.abs(MaxWert);

    // zu ganzer Zahl
    MaxWert=Math.floor(MaxWert);

    if (MaxWert > 0)
    {
        ZufallsZahl=Math.random();    // ab eventuell einschliesslich 0 bis unter 1, Gleitkoma
                                      // eventuell: falls nicht inklusive 0, so doch 0
                                      // letztendlich erzeugbar durch nachfolgendes Runden

        if (ZufallsZahl > 0)
        {
            var Zahler = MaxWert;    // Maxwert bis 0
            var Gefunden = false;
            var Divisor = MaxWert + 1;    // Bsp. Maxwert = 3, also von 0 bis 3 liefern

            do
            {
                if (ZufallsZahl >= (Zahler / Divisor))    //    >= 3/4 so 3
                {    //    >= 2/4 so 2
                    ZufallsZahl = Zahler;    //    >= 1/4 so 1
                    Gefunden = true;    //    >= 0/4 so 0
                }

                Zahler--;    // ab MaxWert bis 0
            }
            while ( (!Gefunden)
                && (Zahler >= 0)
            );
        }
    }

    return ZufallsZahl;
}

```

Beispiel für Ermittlung des größten ganzzahligen Teiler:

```

function AufGanzeZahlPruefen(Wert)
// liefert true wenn ganzzahlig
{return (Math.ceil(Wert) == Wert);} // nächste ganze Zahl über Wert, oder Wert wenn Wert bereits ganz ist

function GanzeZahl_GroesstenGanzzahligenTeilerErmitteln(GanzerWert)
// GanzerWert immer >= 0
// liefert 0 wenn GanzerWert nicht ganzzahlig ist
// liefert sonst ab 1, aber wenn GanzerWert 0 ist so wird 0 geliefert
{
    var ReturnWert = 0;

    // prüfen ob GanzerWert > 0 ist
    if (GanzerWert > 0)

```



```

    {
        // > 0

        // prüfen ob GanzerWert ganzzahlig ist
        if (AufGanzeZahlPruefen(GanzerWert))
        {
            // ist ganzzahlig, also größten ganzzahligen Teiler ermitteln

            var Teiler = 1;
            var GroessterTeiler = 1;

            do
            {
                // nächsten Teiler einstellen
                Teiler++; // ab 2, da ganze Zahl durch 1 immer teilbar ist

                // prüfen ob Gleitkomma-Division eine ganze Zahl ergab
                if (AufGanzeZahlPruefen(GanzerWert / Teiler))
                {
                    // ergab ganze Zahl

                    // prüfen ob neuer Teil größer ist als letzter erkannter Teiler
                    if (GroessterTeiler < Teiler)
                    {
                        // größer als merken
                        GroessterTeiler = Teiler;
                    }
                }
            }
            while (Teiler < (GanzerWert-1));

            // Division durch sich selbst ergibt 1
            // 1 ist aber schon der Initwert
            // von GroessterTeiler

            ReturnWert=GroessterTeiler;
        }
    }

    return ReturnWert;
}

```

Beispiel für Ermittlung des größten gemeinsamen Teiler zweier ganzer Zahlen:

```

function ermittle_ggt_rekursiv(ganze_zahl_1, ganze_zahl_2)    // ganze_zahl_1 >= ganze_zahl_2
{
    if (ganze_zahl_2 == 0)
    {return ganze_zahl_1;}
    else
    {return ermittle_ggt_rekursiv((ganze_zahl_2,(ganze_zahl_1 % ganze_zahl_2));}
}

```

Beispiel für Konvertieren zu ganzzahlig:

```

function KonvertiereZu_GanzeZahl(Wert)
{
    var ReturnWert = 0;

    if (Wert != 0)
    {
        var Faktor = 1;

        if (Wert < 0)
        {
            Faktor = -1;
            Wert = -1 * Wert;

            // Ergebnis ist negativ
            // nur positiven Werten wegen Math.floor()
            // Bsp: 3.1456 wird zu 3
            // also Kommaabschneidung
            // -22,4567 wird zu -23 da -23 < -22
            // also keine Kommaabschneidung
            // aber 22,4567 wird zu 22
            // also Kommaabschneidung
        }
    }
}

```



```

        ReturnWert = Faktor * Math.floor(Wert); // Bsp.: 0,01 wird zu 0
                                                // Bsp.: 1,01 wird zu 1
    }

    return ReturnWert;
}

```

Beispiel für Konvertierung zu ganzzahlig teilbar:

```

function KonvertiereZu_GanzzahligTeilbar(Wert)
// wenn Wert > 0, so liefert die größte ganze Zahl <= Wert, die durch 2 teilbar ist Bsp: 5, so 4 geliefert
// wenn Wert < 0, so liefert die größte ganze Zahl >= Wert, die durch 2 teilbar ist Bsp: -5, so -4 geliefert
{
    // Wert zu ganze Zahl konvertieren, da Modulo den Rest als Gleitkomma liefert
    //      Bsp.: 5 % 2 ergibt Rest 1
    //      4,3 % 2 ergibt nicht Rest 0
    var ReturnWert = KonvertiereZu_GanzeZahl(Wert);

    if (ReturnWert != 0) // Division von 0 durch ReturnWert ungleich Null ergibt immer 0
    {
        if (ReturnWert >= 2) // 1 ist nicht ganzzahlig teilbar
        {
            while ((ReturnWert % 2) != 0)
            {ReturnWert--;} // Bsp.: 5 % 2 Rest 1, dann 4 % 2 Rest 0, also 4 geliefert
        }
        else
        {
            if (ReturnWert <= -2) // -1 ist nicht ganzzahlig teilbar
            {
                while ((ReturnWert % 2) != 0)
                {ReturnWert++;} // Bsp.: -5 % 2 Rest 1, dann -4 % 2 Rest 0, also -4 geliefert
            }
        }
    }

    return ReturnWert;
}

```

Beispiel für ganzzahlige Division:

```

function GanzZahlDivision(Dividend, Divisor) // wenn Divisor == 0, so 0 geliefert
{
    var ReturnWert = 0;

    // Division durch Null ausschliessen
    if (Divisor != 0)
    {
        // Vorzeichen des Ergebnisses ermitteln UND nur positiven Werten wegen Math.floor()
        //      Bsp: 3.1456 wird zu 3 also Kommaabschneidung
        //      -22,4567 wird zu -23 da -23 < -22
        //      also keine Kommaabschneidung
        //      aber 22,4567 wird zu 22 also Kommaabschneidung

        var Faktor = 1;

        if (Dividend < 0)
        {
            Faktor = -1 * Faktor; // Ergebnis ist negativ
            Dividend = -1 * Dividend; // und positiv setzen
        }

        if (Divisor < 0)
        {
            Faktor = -1 * Faktor; // falls Dividend < 0, so Ergebnis positiv; sonst negativ
            Divisor = -1 * Divisor; // und positiv setzen
        }

        ReturnWert = Faktor * Math.floor(Dividend / Divisor);
    }
}

```



```

    return ReturnWert;
}

```

Beispiel für Runden auf n Nachkommastellen:

```

function runden(numerischer_wert, nachkommastellen)
{
    var faktor = Math.pow(10, nachkommastellen);
    return (Math.round(numerischer_wert * faktor) / faktor);
}

```

Eigenschaften:

es wird Gross- und Klein-Schreibweise unterschieden

.E	eulersche Zahl
.LN2	Logarithmus zur Basis 2
.LN10	Logarithmus zur Basis 10
.LOG2E	Logarithmus von E (eulersche Zahl) zur Basis 2
.LOG10E	Logarithmus von E (eulersche Zahl) zur Basis 10
.PI	Zahl Pi
.SQRT1_2	Quadratwurzel von 0,5 also von 1 dividiert durch Quadratwurzel von 2
.SQRT2	Quadratwurzel von 2

Methoden:

.abs(zahl)	Absolutbetrag
.acos(zahl)	Arcus Cosinus im Bogenmass
	liefert Wert von 0 bis PI
.asin(zahl)	Arcus Sinus im Bogenmass
	liefert Wert von -PI/2 bis +PI/2
.atan(zahl)	Arcus Tangens im Bogenmass
	liefert Wert von -PI/2 bis +PI/2
.atan2(zahl1,zahl2)	Arcus Tangens im Bogenmass
	zahl1 y-Position Mittelpunkt
	zahl2 x-Position Mittelpunkt
	liefert Wert von -PI bis +PI
.ceil(zahl)	nächste ganze Zahl oberhalb zahl ermitteln
	Bsp: ceil(1.1) ergibt 2
	ceil(1) ergibt 1
.cos(zahl)	Cosinus im Bogenmass
	liefert Wert von -1 bis +1
.exp(zahl)	liefert den Wert von E hoch zahl
.floor(zahl)	nächste ganze Zahl unterhalb zahl ermitteln
	Bsp: floor(1.1) ergibt 1
	floor(1) ergibt 1
.log(zahl)	liefert den Logarithmus zur Basis E der zahl (natürlicher Logarithmus)
.max(zahl1, zahl2)	größte Zahl aller Zahlen liefern
.min(zahl1, zahl2)	kleinste Zahl aller Zahlen liefern
.pow(basis, exponent)	liefert die Potenz von basis hoch exponent
.random()	Zufallszahl liefern
	0<= zufallszahl <1
.round(zahl)	zahl auf ganz runden; ab >=5 aufwärts
	Bsp: 0,5 ergibt 1
.sin(zahl)	Sinus im Bogenmass
	liefert Wert von -1 bis +1
.sqrt(zahl)	Quadratwurzel ziehen
	zahl >=0
	liefert 0 wenn zahl < 0
.tan(zahl)	Tangens im Bogenmass
	liefert Wert von -1 bis +1

4.2.8. Number Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.
dient zur Instanziierung eines privaten numerischen Objektes
wird verwendet bei numerischen Werten wie z.B. Unendlich

Nicht alle numerischen Methoden sind im Number Script-Objekt implementiert: Es gibt objektübergreifende Methoden, die auch für das Number Script-Objekt zutreffen.

Erzeugung:

```
var Zeiger = new Number(javascript_ausdruck_oder_wert)
```

javascript_ausdruck_oder_wert muss numerisch liefern/sein

Beispiel:

```
var x=Number("three"); // liefert NaN
```



Zugriff:

```
zeiger_auf_number_objekt.eigenschaft
zeiger_auf_number_objekt.methode()
```

```
NUMBER.eigenschaft
NUMBER.methode()
```

Konvertierung nach numerisch:

per Methode Number()

Syntax:

```
[ var Wert = ] Number(Zeiger)

Zeiger    auf zu konvertierende Instanz

Wert      NaN wenn Konvertierung nicht möglich ist
```

Konvertierung von **Hexaziffern** nach numerisch ist **nicht** per Number() möglich:

```
var hexaziffern_feld = [           // anstelle von new Array()
    "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
    "A", "B", "C", "D", "E", "F"
];

function dezimalwert_zu_hexaziffern(dezimal_wert)
{
    var high = Math.floor(dezimal_wert / 16);
    var low  = Math.floor(dezimal_wert - (high * 16));

    return(hexaziffern_feld[high] + hexaziffern_feld[low]);
}
```

Ziffern-Zeichenkettenwert nach numerisch und dabei Dezimalkomma zu Dezimalpunkt umwandeln:

Es wird angenommen, dass genau ein Komma vorkommt.

```
function punkt_zu_komma(zeichenkette)
{
    var pos_komma = zeichenkette.indexOf(",");
    var funktionswert;

    if(pos_komma == -1)
    {
        if (zeichenkette.indexOf(".") == -1)
        { funktionswert = parseInt(zeichenkette); }
        else
        { funktionswert = parseFloat(zeichenkette); }
    }
    else
    {
        funktionswert = parseFloat(
            zeichenkette.substring(0, pos_komma)
            + "."
            + zeichenkette.substring(pos_komma + 1, zeichenkette.length)
        );
    }

    return funktionswert;
}
```

Beispiel für Erzeugung einer Zufallszahl:

```
function ZufallsWertGanzzahligErzeugen1(MaxWert) // Zufallszahl von    inklusive 0
                                                    //                    bis inklusive MaxWert
                                                    //                    erzeugen als ganze Zahl
                                                    // liefert 0 wenn MaxWert 0 ist

// MaxWert:    wenn Gleitkomma, so vor Verwendung nach ganzzahlig konvertiert
//              wenn negativ, so Absolutwert verwendet und wenn dann Gleitkomma, dann zu ganzer Zahl
//              Bsp.:    -22.5 wird zu 22.5 wird zu 22
//              jedoch NICHT -22.5 wird zu -23 wird zu 23

{
    var ZufallsZahl=0;
```




```

// zu positiv
MaxWert=Math.abs(MaxWert);

// zu ganzer Zahl
MaxWert=Math.floor(MaxWert);

if (MaxWert > 0)
{
    ZufallsZahl=Math.random();    // ab eventuell einschliesslich 0 bis unter 1, Gleitkomma
                                   // eventuell: falls nicht inklusive 0, so doch 0
                                   // letztendlich erzeugbar durch nachfolgendes Runden

    if (ZufallsZahl > 0)           // Bps. MaxWert 3 Zufallszahl   0,412345
                                   //                               0,612345
                                   //                               0,0015545
                                   //                               0,29456
    {
        // Kommaverschiebung
        while ((ZufallsZahl * 10) <= MaxWert)
        {ZufallsZahl = ZufallsZahl * 10;}    // 0,412345  0,612345 1,5545 2,9456

        ZufallsZahl= Math.round(ZufallsZahl);    // Kommastellen abschneiden durch runden
                                                  // ab inklusive 0,5 aufwärts, sonst abwärts
                                                  // 0          1          2          3

        // falls Runden einen Wert > MaxWert ergab, so den Wert begrenzen auf MaxWert
        if (ZufallsZahl > MaxWert)
        {ZufallsZahl = MaxWert;}
    }
}

return ZufallsZahl;
}

function ZufallsWertGanzzahligErzeugen(MaxWert) // Zufallszahl von    inklusive 0
                                                  //                               bis inklusive MaxWert
                                                  //                               erzeugen als ganze Zahl
                                                  // liefert 0 wenn MaxWert 0 ist
// MaxWert:    wenn Gleitkomma, so vor Verwendung nach ganzzahlig konvertiert
//             wenn negativ, so Absolutwert verwendet und wenn dann Gleitkomma, dann zu ganzer Zahl
//             Bsp.:    -22.5 wird zu 22.5 wird zu 22
//             jedoch NICHT -22.5 wird zu -23 wird zu 23
{
    var ZufallsZahl=0;

    // zu positiv
    MaxWert=Math.abs(MaxWert);

    // zu ganzer Zahl
    MaxWert=Math.floor(MaxWert);

    if (MaxWert > 0)
    {
        ZufallsZahl=Math.random();    // ab eventuell einschliesslich 0 bis unter 1, Gleitkomma
                                       // eventuell: falls nicht inklusive 0, so doch 0
                                       // letztendlich erzeugbar durch nachfolgendes Runden

        if (ZufallsZahl > 0)
        {
            var Zahler = MaxWert;    // Maxwert bis 0
            var Gefunden = false;
            var Divisor = MaxWert + 1;    // Bsp. Maxwert = 3, also von 0 bis 3 liefern

            do
            {
                if (ZufallsZahl >= (Zahler / Divisor))    // >= 3/4 so 3
                {    // >= 2/4 so 2
                    ZufallsZahl = Zahler;    // >= 1/4 so 1
                    Gefunden = true;    // >= 0/4 so 0
                }
            }
        }
    }
}

```



```

        Zahler--; // ab MaxWert bis 0
    }
    while ( (!Gefunden)
        && (Zahler >= 0)
    );
}

return ZufallsZahl;
}

```

Beispiel für Ermittlung des größten ganzzahligen Teiler:

```

function AufGanzeZahlPruefen(Wert)
// liefert true wenn ganzzahlig
{return (Math.ceil(Wert) == Wert);} // nächste ganze Zahl über Wert, oder Wert wenn Wert bereits ganz ist

function GanzeZahl_GroesstenGanzzahligenTeilerErmitteln(GanzerWert)
// GanzerWert immer >= 0
// liefert 0 wenn GanzerWert nicht ganzzahlig ist
// liefert sonst ab 1, aber wenn GanzerWert 0 ist so wird 0 geliefert
{
    var ReturnWert = 0;

    // prüfen ob GanzerWert > 0 ist
    if (GanzerWert > 0)
    {
        // > 0

        // prüfen ob GanzerWert ganzzahlig ist
        if (AufGanzeZahlPruefen(GanzerWert))
        {
            // ist ganzzahlig, also größten ganzzahligen Teiler ermitteln

            var Teiler = 1;
            var GroessterTeiler = 1;

            do
            {
                // nächsten Teiler einstellen
                Teiler++; // ab 2, da ganze Zahl durch 1 immer teilbar ist

                // prüfen ob Gleitkomma-Division eine ganze Zahl ergab
                if (AufGanzeZahlPruefen(GanzerWert / Teiler))
                {
                    // ergab ganze Zahl

                    // prüfen ob neuer Teil größer ist als letzter erkannter Teiler
                    if (GroessterTeiler < Teiler)
                    {
                        // größer als merken
                        GroessterTeiler = Teiler;
                    }
                }
            }
            while (Teiler < (GanzerWert-1)); // Division durch sich selbst ergibt 1
                                           // 1 ist aber schon der Initwert
                                           // von GroessterTeiler

            ReturnWert=GroessterTeiler;
        }
    }

    return ReturnWert;
}

```

Beispiel für Ermittlung des größten gemeinsamen Teiler zweier ganzer Zahlen:

```

function ermittle_ggt_rekursiv(ganze_zahl_1, ganze_zahl_2) // ganze_zahl_1 >= ganze_zahl_2
{
    if (ganze_zahl_2 == 0)

```



```

    {return ganze_zahl_1;}
    else
    {return ermittle_ggt_rekursiv((ganze_zahl_2,(ganze_zahl_1 % ganze_zahl_2));)}
}

```

Beispiel für Konvertieren zu ganzzahlig:

```

function KonvertiereZu_GanzeZahl(Wert)
{
    var ReturnWert = 0;

    if (Wert != 0)
    {
        var Faktor = 1;

        if (Wert < 0)
        {
            Faktor = -1;           // Ergebnis ist negativ
            Wert  = -1 * Wert;      // nur positiven Werten wegen Math.floor()
                                   // Bsp: 3.1456 wird zu 3
                                   // also Kommaabschneidung
                                   // -22,4567 wird zu -23 da -23 < -22
                                   // also keine Kommaabschneidung
                                   // aber 22,4567 wird zu 22
                                   // also Kommaabschneidung
        }

        ReturnWert = Faktor * Math.floor(Wert); // Bsp.: 0,01 wird zu 0
                                                // Bsp.: 1,01 wird zu 1
    }

    return ReturnWert;
}

```

Beispiel für Konvertierung zu ganzzahlig teilbar:

```

function KonvertiereZu_GanzzahligTeilbar(Wert)
// wenn Wert > 0, so liefert die größte ganze Zahl <= Wert, die durch 2 teilbar ist Bsp: 5, so 4 geliefert
// wenn Wert < 0, so liefert die größte ganze Zahl >= Wert, die durch 2 teilbar ist Bsp: -5, so -4 geliefert
{
    // Wert zu ganze Zahl konvertieren, da Modulo den Rest als Gleitkomma liefert
    // Bsp.: 5 % 2 ergibt Rest 1
    // 4,3 % 2 ergibt nicht Rest 0
    var ReturnWert = KonvertiereZu_GanzeZahl(Wert);

    if (ReturnWert != 0) // Division von 0 durch ReturnWert ungleich Null ergibt immer 0
    {
        if (ReturnWert >= 2) // 1 ist nicht ganzzahlig teilbar
        {
            while ((ReturnWert % 2) != 0)
            {ReturnWert--;} // Bsp.: 5 % 2 Rest 1, dann 4 % 2 Rest 0, also 4 geliefert
        }
        else
        {
            if (ReturnWert <= -2) // -1 ist nicht ganzzahlig teilbar
            {
                while ((ReturnWert % 2) != 0)
                {ReturnWert++;} // Bsp.: -5 % 2 Rest 1, dann -4 % 2 Rest 0, also -4 geliefert
            }
        }
    }

    return ReturnWert;
}

```

Beispiel für ganzzahlige Division:

```

function GanzZahlDivision(Dividend, Divisor) // wenn Divisor == 0, so 0 geliefert
{
    var ReturnWert = 0;

```



```

// Division durch Null ausschliessen
if (Divisor != 0)
{
    // Vorzeichen des Ergebnisses ermitteln UND nur positiven Werten wegen Math.floor()
    //      Bsp:      3.1456  wird zu 3      also Kommaabschneidung
    //      -22,4567 wird zu -23 da -23 < -22
    //      also keine Kommaabschneidung
    //      aber 22,4567 wird zu 22 also Kommaabschneidung

    var Faktor = 1;

    if (Dividend < 0)
    {
        Faktor = -1 * Faktor;      // Ergebnis ist negativ
        Dividend = -1 * Dividend;  // und positiv setzen
    }

    if (Divisor < 0)
    {
        Faktor = -1 * Faktor; // falls Dividend < 0, so Ergebnis positiv; sonst negativ
        Divisor = -1 * Divisor;      // und positiv setzen
    }

    ReturnWert = Faktor * Math.floor(Dividend / Divisor);
}

return ReturnWert;
}

```

Beispiel für Runden auf n Nachkommastellen:

```

function runden(numerischer_wert, nachkommastellen)
{
    var faktor = Math.pow(10, nachkommastellen);
    return (Math.round(numerischer_wert * faktor) / faktor);
}

```

Eigenschaften:

es wird Gross- und Kleinschreibung unterschieden

MAX_VALUE	<p>maximaler numerischer endlicher Wert > 0 oder < 0, den Javascript zulässt: 1.79E+308</p> <p>Werte über MAX_VALUE sind unendlich (Number.POSITIVE_INFINITY bzw. Number.NEGATIVE_INFINITY)</p> <p>kann als Wert zugewiesen werden</p> <p>nur lesen</p>						
MIN_VALUE	<p>minimaler numerischer endlicher Wert > 0, den Javascript zulässt: 5E-324</p> <p>Werte unter MIN_VALUE sind immer 0</p> <p>kann als Wert zugewiesen werden</p> <p>nur lesen</p>						
NaN	<p>nicht numerischer Wert (Not a Number)</p> <p>kann als Wert zugewiesen werden</p> <p>Bsp: var Wert = Number.NaN;</p> <p>Hinweis: Methoden .parseFloat() und .parseInt() liefern NaN, wenn Parameter nicht numerisch ist.</p> <p>Vergleich mit NaN ist nicht zulässig: Dafür ist die Methode .isNaN() zu verwenden.</p> <p>nur lesen</p>						
NEGATIVE_INFINITY	<p>entspricht negativ unendlich</p> <p>kann als Wert zugewiesen werden</p> <p>Hinweis für numerische Operationen</p> <table> <tr> <td>Multiplikation</td><td> <p>Wert mal unendlich ergibt unendlich</p> <p>unendlich mal unendlich ergibt unendlich</p> <p>0 * unendlich ergibt NaN</p> <p>NaN * unendlich ergibt NaN</p> </td></tr> <tr> <td>Division</td><td> <p>Wert durch unendlich ergibt 0</p> <p>unendlich durch Wert ergibt unendlich</p> <p>unendlich durch unendlich ergibt NaN</p> <p>Division durch 0 nicht erlaubt</p> </td></tr> <tr> <td>Vorzeichen</td><td> <p>wird wie üblich ermittelt</p> </td></tr> </table> <p>nur lesen</p>	Multiplikation	<p>Wert mal unendlich ergibt unendlich</p> <p>unendlich mal unendlich ergibt unendlich</p> <p>0 * unendlich ergibt NaN</p> <p>NaN * unendlich ergibt NaN</p>	Division	<p>Wert durch unendlich ergibt 0</p> <p>unendlich durch Wert ergibt unendlich</p> <p>unendlich durch unendlich ergibt NaN</p> <p>Division durch 0 nicht erlaubt</p>	Vorzeichen	<p>wird wie üblich ermittelt</p>
Multiplikation	<p>Wert mal unendlich ergibt unendlich</p> <p>unendlich mal unendlich ergibt unendlich</p> <p>0 * unendlich ergibt NaN</p> <p>NaN * unendlich ergibt NaN</p>						
Division	<p>Wert durch unendlich ergibt 0</p> <p>unendlich durch Wert ergibt unendlich</p> <p>unendlich durch unendlich ergibt NaN</p> <p>Division durch 0 nicht erlaubt</p>						
Vorzeichen	<p>wird wie üblich ermittelt</p>						



POSITIVE_INFINITY	entspricht positiv unendlich kann als Wert zugewiesen werden Hinweis für numerische Operationen
Multiplikation	Wert mal unendlich ergibt unendlich unendlich mal unendlich ergibt unendlich 0 * unendlich ergibt NaN NaN * unendlich ergibt NaN
Division	Wert durch unendlich ergibt 0 unendlich durch Wert ergibt unendlich unendlich durch unendlich ergibt NaN Division durch 0 nicht erlaubt
Vorzeichen	wird wie üblich ermittelt
nur lesen	

Methoden:

Soll ein Literal verwendet werden, das **nicht** mindestens 1 Dezimalkomma enthält (Dezimalpunkt ist das Komma), so muss unmittelbar vor .methode() 1 Blank kodiert werden, damit das Literal nicht als Bezeichner einer Referenz interpretiert wird.

Beispiel: 77.77.toExponential() liefert "7.777e+1"
 77 .toExponential(); liefert "7.7e+1"
 77.toExponential(); liefert Syntaxfehler
 1250 .toPrecision(2) liefert "1.3e+3"

.toExponential() String liefern, der den Wert in Exponential-Darstellung enthält
 auch bei numerischem Literal z.B. "10.2345678e+13"
 IE ab 5.5, NS ab 6.x

Beispiel: var num=77.1234
 num.toExponential() liefert "7.71234e+1"
 num.toExponential(4) liefert "7.7123e+1"
 num.toExponential(2) liefert "7.71e+1"
 77.1234.toExponential() liefert "7.71234e+1"
 77 .toExponential() liefert "7.7e+1"

.toFixed() String liefern, der den Wert in Festkomma-Darstellung enthält
 auch bei numerischem Literal z.B. "10.2345678"
 IE ab 5.5, NS ab 6.x

Beispiele:
 var num=10.1234
 num.toFixed() liefert "10"
 num.toFixed(4) liefert "10.1234"
 num.toFixed(2) liefert "10.12"
 0.124.toFixed(2) liefert "0.12"
 0.125.toFixed(2) liefert "0.13"
 0.126.toFixed(2) liefert "0.13"
 0.045.toFixed(2) liefert "0.05"
 1234.56789.toFixed(4) liefert "1234.5679"

.toPrecision() liefert String, der die Nachkommastellen enthält
 auch bei numerischem Literal z.B. "10.2345678"
 IE ab 5.5, NS ab 6.x

Beispiele:
 var num=5.123456
 num.toPrecision() liefert "5.123456"
 num.toPrecision(4) liefert "5.123"
 num.toPrecision(2) liefert "5.1"
 num.toPrecision(1) liefert "5"
 1250 .toPrecision(2) liefert "1.3e+3"
 1250 .toPrecision(5) liefert "1250.0"
 1234.56789.toPrecision(2) liefert "1.2e+3"
 1234.56789.toPrecision(9) liefert "1.234.56789" entspricht also e+0

objektübergreifende Methoden beim Number Script-Objekt:

isFinite() ermittelt, ob Wert einer Instanz numerisch endlich ist
 Syntax:

[var Wert =] isFinite(Zeiger);

Zeiger auf Instanz vom Number-Objekt oder Ausdrucksergebnis
 auch Number.POSITIVE_INFINITY
 Number.NEGATIVE_INFINITY
 Number.NaN



	Wert	true, so Wert der Instanz ist endlich false, so Wert der Instanz ist unendlich immer bei Number.POSITIVE_INFINITY Number.NEGATIVE_INFINITY Number.NaN
isNaN()	ermittelt, ob Wert einer Instanz nicht numerisch ist Syntax:	[var Wert =] isNaN(Zeiger);
	Zeiger	auf Instanz vom Number-Objekt oder Ausdrucksergebnis auch Number.POSITIVE_INFINITY Number.NEGATIVE_INFINITY Number.NaN
	Wert	true, so Wert der Instanz ist nicht numerisch immer bei Number.NaN false, so Wert der Instanz ist numerisch immer bei Number.POSITIVE_INFINITY Number.NEGATIVE_INFINITY
Beispiele:		
<pre>var Kette ="10.23"; var Wert =10.23; var Ergebnis1=parseFloat(Kette); var Ergebnis2=floatValue=parseFloat(Wert); alert(isNaN(Ergebnis1); alert(isNaN(Ergebnis2);</pre>		
Hinweis: Methoden parseFloat und parseInt liefern NaN, wenn Parameter nicht numerisch ist		
Bsp: var zahl=parseFloat("text"); ifNaN(zahl) liefert true		
Number()	konvertiert eine Instanz zu einem numerischen Wert (Number-Objekt-Wert) Syntax:	[var Wert =] Number(Zeiger)
	Zeiger 1	auf zu konvertierende Instanz auch Date-Objekt
	Wert	wenn Zeiger 1 ein Date-Objekt ist, so Wert ist die Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr Weltzeit (UTC), wobei Weltzeit genau GMT entspricht wenn Datum vor dem 1.1.1970 0 Uhr Weltzeit so Wert negativ NaN wenn Konvertierung nicht möglich ist
Beispiel:		
<pre>var DatumJetzt = new Date(); alert (Number(DatumJetzt));</pre>		
parseFloat()	String oder Literal	parsen und nach Floating-point umwandeln können enthalten Vorzeichen + und - Ziffern 0 bis 9 Dezimalkomma als Punkt e oder E Blanks (werden ignoriert) falls andere Zeichen enthalten sind, gilt: String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann konvertiert
Erfolg der Konvertierung ist per Methode isNaN() zu prüfen		
Syntax:		
[var Wert =] parseFloat(Kette)		
	Kette	String oder Literal
	Wert	Floating-point NaN wenn String bzw. Literal bereits mit erstem Zeichen fehlerhaft
Beispiele:		



gültiges Literal
 parseFloat("3.14")
 parseFloat("314e-2")
 parseFloat("0.0314E+2")

gültiger String
 var x = "3.14"
 parseFloat(x)

ungültiges Literal
 parseFloat("FF2")

parseInt() String oder Literal parsen und nach Integer umwandeln
 können enthalten
 Vorzeichen + und -
 Ziffern 0 bis 9, aber keine Vornull(en)
 Blanks (werden ignoriert)
 eventuelle Buchstaben A bis F

 falls andere Zeichen enthalten sind, gilt:
 String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann
 konvertiert
 Erfolg der Konvertierung ist per Methode isNaN() zu prüfen
 Syntax:
 [var Wert2 =] parseInt(Kette[, Wert1])

 Kette String oder Literal
 muss passend zu Wert1 kodiert sein
 Hinweis: oktale Escape-Sequenzen sind in ab Javascript 1.5 im
 Netscape 6.x deprecated

 Wert1 Integer
 Zahlenbasis
 10 für dezimal
 8 für oktal
 16 für hexadezimal
 (Wert1 kann Buchstaben A bis F enthalten)
 Standard: 0

 Wert2 Integer
 NaN wenn String bzw. Literal bereits mit erstem
 Zeichen fehlerhaft
 wenn Wert1 0 ist, so
 wenn Kette beginnt mit
 "0x", so hexadezimale Konvertierung
 "0" und nicht "0x", so oktale Konvertierung
 nicht mit "0" oder "0x", so dezimale Konvertierung

Beispiel:

gültige Literale:
 parseInt("F", 16)
 parseInt("17", 8)
 parseInt("15", 10)
 parseInt(15.99, 10)
 parseInt("FXX123", 16)
 parseInt("1111", 2)
 parseInt("15*3", 10)
 parseInt("17")
 parseInt("0x7", 16)
 parseInt("0x7")
 parseInt("0")
 parseInt("0x11", 16)
 parseInt("0x11", 0)
 parseInt("0x11")

ungültige Literale:
 parseInt("F")
 parseInt("Hello", 8)
 parseInt("0x7", 10)
 parseInt("FFF", 10)
 parseInt("002")



`.toString()` String liefern, der den Wert enthält, wobei Nachkommastellen automatisch festgelegt werden und gerundet wird
auch bei numerischem Literal z.B. "10.2345678"
siehe Script-Objekt Number
Syntax:
[var Kette =] zeiger_auf_number_objekt.toString([Wert])

Wert2 Integer, 2 bis 36
Zahlenbasis
10 für dezimal
8 für oktal
16 für hexadezimal
(Wert1 kann Buchstaben A bis F enthalten)
wenn nicht kodiert, so automatische Festlegung
wenn Kette beginnt mit
"0x", so hexadezimale Basis
"0" und nicht "0x", so oktale Basis
nicht mit "0" oder "0x", so dezimale Basis

Beispiel

```
var howMany=10;
howMany.toString() liefert "10"
45.toString() liefert "45"
```

`.valueOf()` numerischen Wert liefern
auch bei numerischem Literal z.B. "10.2345678"
siehe Script-Objekt Number
Syntax:
[var Kette =] zeiger_auf_number_objekt.valueOf()

4.2.9. Objekt JScript-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache und stellt das Basis-Objekt aller JScript-Objekte (außer seiner selbst) dar. Es repräsentiert ein JScript-Objekt schlechthin und ohne speziellen Datentyp. Es repräsentiert **nicht** das Objekt object im Browser, welches das HTML-Element OBJECT widerspiegelt.

JScript-Objekte gibt es mit spezialisierten Objektklassen (Objekttypen), die einen speziellen Datentyp verwalten,

z.B.

- Objekt arguments
- Objekt Array
- Objekt Boolean
- Objekt Date
- Objekt Enumerator
- Objekt Error
- Objekt Function
- Objekt Math
- Objekt Number
- Objekt object (nicht Objekt object des Internet Explorer für das HTML-Tag OBJECT)
- Objekt RegExp
- Objekt String
- Objekt var

Anhand dieser speziellen JScript-Objekte sind Beschreibungen von Daten möglich.

Das Script-Objekt object umfasst also alle Eigenschaften und Methoden, die in **fast allen** JScript-Objekten und deren Instanzen implementiert sind.

Ein Instanz eines JScript-Objektes kann per Anweisung new erzeugt werden (siehe dort):

Achtung: Der Browser kann nur Objekte verarbeiten, die er kennt, z.B. ein Array-Objekt, dessen Methoden und Eigenschaften dem Browser bekannt sind. Bei einem privaten Objekt müssen alle Eigenschaften und Methoden auf Script-Komponenten bestehen.

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft `.prototype` erzeugt! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft `.prototype`.
Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft `.prototype`.

Punktnotation zum Zeiger:
zeiger.**prototype**.eigenschaft
zeiger.**prototype**.methode
Erweiterung eines Script-Objektes per
bezeichner_script_objekt.**prototype**.eigenschaft



bezeichner_script_objekt.prototype.methode
Eigenschaften und Methoden müssen auf Script-Elemente **basieren**, denn nur letztere kennt die Scriptmaschine des Browsers.

Eine Referenz auf eine Objekt-Instanz ist auch per Anweisungen this und with möglich (siehe dort).

Prinzipiell dürften andere Scripthersteller wegen der Script-Kompatibilität die gleichen Script-Objekte und deren Eigenschaften und Methoden wie in JScript unterstützen.

Eigenschaften:

.constructor Bezeichner einer JScript-Objektklasse (Objekttyp) oder eines privaten Konstruktors
Anwendung: Ermittlung der Objektklasse/Konstruktors eines abgeleiteten Objektes
Ableitung aus der Objektklasse
per Anweisung new mit der Objektklasse als Konstruktor benötigt (siehe dort)
nicht bei Script-Objekt Math möglich
Ableitung aus privatem Konstruktor
per Anweisung new, die den privaten Konstruktor verwendet
JScript-Objektklassen sind z.B.
Array
Boolean
Date
Function

Beispiel 1 für Ableitung aus einem JScript-Objekt:

```
var Kette = new String("Hi");
alert( (Kette.constructor == String)); // liefert "true"
alert( (Kette.constructor == "String")); // liefert "false"
```

Beispiel 2 für Ableitung anhand privaten Konstruktors:

```
function TestFunktion()
{ alert("Hallo"); }

var ZeigerAufFunktion = new TestFunktion(); // bewirkt Ausführung von TestFunktion() also auch von alert()

// ZeigerAufFunktion(); // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
//                       // da keine Ableitung vom JScript-Objekt Function
// Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration erkannt

alert(ZeigerAufFunktion.constructor == TestFunktion); // true
alert(TestFunktion.constructor == TestFunktion); // false
```

.propertyIsEnumerable prüfen ob Stringwert in einem Objekt als Menge von String-Elementen enthalten ist
und ob das Objekt mit der Anweisung for in verarbeitet werden kann

Beispiel:

```
var Feld = new Array("Apfel", "Banane", "Zitrone");
alert( (Feld.propertyIsEnumerable("Apfel")); // true
```

.prototype Zeiger auf den Prototyp-Bereich im Objekt, der per Prototyping erweitert wird
Objekt ist entweder Script-Objekt oder bereits instanziiertes Objekt:
innerhalb eines Konstruktors ist .prototype nicht zu kodieren
Prototyping eines Objektes verändert den Umfang der Standard-Methoden und -Eigenschaften zum Objekt zur Laufzeit der Scriptmaschine.
Script-Objekte können nur erweitert werden.
Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype .
Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft .prototype .
siehe auch .isPrototypeOf() und .hasOwnProperty()

Beispiel für Erweiterung des Script-Objektes Array, das die Eigenschaft .prototype besitzt:

```
function MaximumErmittleIn ( )
{
    var max = this[0]; // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                     // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }
}
```



```

    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmitteln; // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6); // 6 numerische Elemente

// neue Methode des JScript-Objektes Array aufrufen
alert(Feld.NeueArrayMethode());

```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
Es wird die Eigenschaft .prototype **nicht** erzeugt !

```

<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
    this.nummer = nummer;
    this.plz = plz;
    this.ort = ort;
    this.methode = methode;
}

person = new datenstruktur_erzeugen
(
    "Erika",           // Vorname
    "Mustermann",     // Nachname
    "Musterstrasse",  // Strasse
    "1",              // Nummer
    "10000",          // PLZ
    "Musterstadt",    // Ort
    datenstruktur_ausgeben // ohne () kodieren
);

// alternativ auch kodierbar:
// var person = {
//     vorname:"Erika",           // Vorname
//     nachname:"Mustermann",    // Nachname
//     strasse:"Musterstrasse",  // Strasse
//     nummer:"1",              // Nummer
//     plz:"10000",             // PLZ
//     ort:"Musterstadt",       // Ort
//     methode:datenstruktur_ausgeben// Ausgabemethode ohne () kodieren
// };

// Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

```



```
alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !
// -->
</SCRIPT>
</HTML>
```

Methoden:

escape()

kodiert einen String oder ein Literal in das Unicode-Format
 URI (Uniform Resource Identifiers) werden nicht kodiert
 Unicode-Format: kann von jedem Browser verarbeitet werden
 folgende Zeichen werden dargestellt im Format %XX
 Leerzeichen
 Punkt, Komma etc. (alle Satzzeichen)
 nicht ASCII-Zeichen
 XX ist Hexadezimal-Ziffernfolge
 Bsp.: Leerzeichen als %20
 folgende Zeichen werden dargestellt im Format %uXXXX
 Zeichen des erweiterten Zeichensatzes ab inklusive 256
 XX ist Hexadezimal-Ziffernfolge

Beispiel für Einbindung einer Suchmaschine:

```

var markierter_text=document.selection.createRange().text;
        // oder var markierter_text=prompt('Suchbegriff: ');
var suchmaschinen_url='http:// .....';
var suchmaschinen_parameter=':.....';

if (markierter_text)
{location.HREF=suchmaschinen_url + suchmaschinen_parameter + escape(markierter_text);}
else
{location.HREF=suchmaschine_url; }

```

```
Beispiel für altavista: suchmaschinen_url      'http://altavista.de/'
                        suchmaschinen_parameter 'cgi-bin/query?pg=q&what=web&q='
```

eval()	parsen und, falls fehlerfrei, sofortiges Ausführen eines Scriptcodes
--------	--

Beispiel 1:

```

if (VarUserAgent != "")
{
    for ( var i=0; i < 10; i++)
    {
        eval(      'if (VarUserAgent.indexOf("'" + i + '."') != -1)'
                  + '{VarBrowser_Version_Haupt = ' + i + '};'
                  );
    }
}

```

Beispiel 2:

```
function DatumHolen(Zeiger)
{
    var Kette = "Heute ist: ";
    Kette += Zeiger.getDate() + "/";
    Kette += (Zeiger.getMonth() + 1) + "/";
    Kette += Zeiger.getYear();

    return(Kette);
}
```

```
var Kette="Date":
```

```
eval("var Jetzt = new " + Kette + "");// zur Laufzeit erzeugen
    "alert(DatumHolen(Jetzt));"      // Parameter ist zur Laufzeit bekannt
);
```

```
.hasOwnProperty()
```

prüfen ob zum Objekt eine Eigenschaft vorhanden ist, jedoch leider **nicht** aus Prototyping einer per new erzeugten Instanz des JScript-Objektes

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype .

siehe auch .prototype

Beispiel:

```
function MaximumErmitteln()
{
    var max = this[0]; // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist.
```



```
// also Variable Feld

for (var i = 1; i < this.length; i++)
{
    if (max < this[i])
    {max = this[i];}
}

return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmitteln; // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6); // 6 numerische Elemente
// Es wird die Eigenschaft .prototype erzeugt.

alert( (Feld.prototype.hasOwnProperty("NeueArrayMethode")); // leider false
alert( (Array.prototype.hasOwnProperty("NeueArrayMethode ")); // true
```

.isPrototypeOf() prüfen ob Objekt per new erzeugt wurde, also eine Instanz eines anderen JScript-Objektes ist
Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype .
siehe auch .prototype

Beispiel:

```
var Variable = new RegExp();
alert( (RegExp.prototype.isPrototypeOf(Variable)); // true.
```

.toLocaleString() Wert eines Objektes in System-lokale Einstellungen umwandeln
System-lokale Einstellung z.B. Ländereinstellung, Uhrzeitformat
Konvertierung: Analog wie z.B. bei Excel, das die lokalen Einstellungen ausliest.
nur anwenden für Anzeige des konvertierten Wertes:
Berechnungen mit dem Wert immer unkonvertiert vollziehen, da sämtliche
Berechnungsfunktionen **nur** das interne, also unkonvertierte
Format kennen (**nicht** analog zu Excel)
Wenn das Objekt ein Feld ist, also Feldelemente in lokale Einstellungen konvertiert werden sollen,
dann wird ein String geliefert, der die Feldelemente in der Reihenfolge im Feld
und diese getrennt durch **dasjenige** Trenner-Zeichen laut **lokale** Einstellungen enthält.
Wenn das Objekt eine Date-Objekt ist, so wird der Wert von dem Objekt in den lokalen
Datumeinstellungen geliefert. Dabei sind nur Jahreszahlen von 1601 bis 9999 zulässig.
Andere Jahresangaben werden nicht konvertiert.
Wenn das Objekt ein Number-Objekt ist, so werden die lokalen Einstellungen zum Zahlenformat
geliefert.
Wenn das Objekt ein JScript-Objekt object ist, so werden nur dann lokale Einstellungen berücksichtigt,
insoweit das Objekt diese berücksichtigen kann. Ein String wird immer geliefert.

Beispiel für Konvertierung eines Feld mit Gleitkomma-Werten:

```
var Feld = new Array(6);
var Wert = 3,201,300.20; // in JScript ist das Dezimalkomma der Punkt
// der Tausendertrenner das Komma

// Feld füllen
for( var i = 0; i < 7; i++)
{
    Wert +1 = 10;
    Feld [i] = Wert;
}

alert(Feld.toLocaleString());
```

Beispiel für Konvertierung eines Datums:

```
var Jetzt = new Date();
alert(Jetzt.toLocaleString());
```

toString() Wert eines Objektes in einen String umwandeln
wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma getrennt
geliefert



wenn Objekt ein Script-Objekt Boolean ist, dann "true" bzw "false" geliefert (Kleinschreibung !)
 wenn Objekt ein Script-Objekt Error ist, dann die Fehlermeldung geliefert (Objekt-Wert ist der Fehlercode)
 wenn Objekt ein Script-Objekt Function ist, dann Quellcode der Funktion geliefert
 (inklusive Funktionskopf)

wenn Objekt ein Script-Objekt Object ist, so wird geliefert:
 "[**object** objectname]"
 mit objectname als konkreter Bezeichner der Objektklasse bzw. des Objektes
 fettgedrucktes wird so geliefert wie angegeben

wenn der Bezug vor .toString() ein Wert laut ID-Attribut oder NAME-Attribut ist, dann wird die
 Objektklasse geliefert

Beispiel 1:

```
var Wert = 33,33;
alert(Wert.toString(2) + ' ' + Wert.toString(16) + ' ' + Wert.toString(10));
```

Beispiel 2:

```
<BUTTON ID="ID_Button" .... > .... </BUTTON>
ID_Button.toString() liefert "button"
```

unescape Dekodiert einen per .escape kodierten String in einen normalen String
 URI (Uniform Resource Identifiers) sind nicht per escape()kodierbar

valueOf() Wert eines JScript-Objektes bzw. Instanz eines JScript-Objektes ermitteln
 nicht bei Script-Objekt Math und Script-Objekt Error (auch nicht bei Instanzen dieser Objekte)
 Wert ist im Datentyp des Objektes, aber:

- wenn Objekt ein Array-Objekt ist, dann Elemente in der Feldreihenfolge und mit Komma
 getrennt geliefert (identisch in der Wirkung mit .toString() und der Array-Methode
 .join())
- wenn Objekt ein Boolean Objekt ist, dann true bzw false geliefert (kein String !)
- wenn Objekt ein Date Objekt ist, dann Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr
 geliefert
- wenn Objekt ein Function Objekt ist, dann Zeiger auf Funktion geliefert
- wenn Objekt ein Number Script-Objekt ist, dann numerischen Wert geliefert
- wenn Objekt ein Script-Objekt object ist, so Zeiger geliefert

siehe auch .toLocaleString() und .toString()

4.2.10. String Script-Objekt

Dieses Objekt ist eine Komponente der Scriptsprache.
 entspricht einer Zeichenkettenvariable
 Zeichenkettenkonstante (String-Literal)
 ist gleichzeitig Feld der Zeiger aller Zeichenelemente im String:
 Index-Operationen beim IE nicht möglich, da Collection und nicht reines Feld

Bsp: document.writeln("Dieser Text ist schmal".small);

Erzeugung:

```
var Zeiger = new String(String-Literal oder Variable oder Ausdruck);
var Zeiger = String-Literal;
```

String-Literal	alphanumerische Zeichen in " " oder ' ' kodieren
Variable	muss String sein
Ausdruck	muss String liefern

Methode .toString()
 .valueOf()
 String()
 Methoden des Script-Objektes Date

Zugriff:

```
zeiger_auf_string_objekt.eigenschaft
zeiger_auf_string_objekt.methode()
zeiger_auf_string_objekt[Index]
```

zeiger_auf_string_objekt [Index]	laut Erzeugung nicht beim IE [] muss kodiert werden Index Integer ab 0 bis .length -1
-------------------------------------	--

```
"wert_aus_zeichen".eigenschaft
"wert_aus_zeichen".methode()
"wert_aus_zeichen"[Index]
```

"wert_aus_zeichen" Stringliteral



	[Index]	nicht beim IE [] muss kodiert werden Index Integer ab 0 bis .length -1
Index-Zugriff	nicht beim IE Beispiel: <code>var Kette = "Hello"; alert(Kette[0]); // "H"</code>	
String-Literal	wird wie String-Objekt verwaltet: Eigenschaften und Methoden des String-Objektes ebenfalls nutzbar, wobei Methoden zum String-Literal, die Zeiger liefern, dann nur Zeiger auf ein String-Literal liefern und nicht auf einen String-Objekt dient der vereinfachten Kodierung vorallem der Parameterbelegung von eval() per Ausdruck: Wenn Parameter von eval ein Zeiger auf ein String-Objekt ist, so erkennt eval() keinen Ausdruck also z.B. keine Operatoren etc. ! Beispiele: <code>var StringLiteral = "2 + 2"; var StringObjekt = new String("2 + 2"); eval(StringLiteral); // liefert Zeiger auf numerische Variable mit Wert 4 eval("2+2"); // liefert Zeiger auf numerische Variable mit Wert 4 eval(StringObjekt); // liefert Kette "2 + 2"</code>	
Eigenschaften:		
.length	Anzahl der Zeichen im String bzw. String-Literal Integer, ab 0 wenn 0 so Leerkette Syntax: <code>[var Wert =] zeiger_auf_string_objekt.length</code> Beispiel: <code>var StringLiteral = "StringLiteral"; StringLiteral.length // liefert 13 "StringLiteral".length // liefert 13</code>	zeiger_auf_string_objekt auch direkt kodiertes String-Literal
Methoden:		
stellen die Stringoperationen dar siehe auch Operator + für Verkettung Hinweis:	.toString()	liefert Zeiger auf Wert des String bzw. Stringliteral Syntax: <code>[var Zeiger =] zeiger_auf_string_objekt.toString()</code> zeiger_auf_string_objekt auch direkt kodiertes String-Literal
	valueOf()	liefert Zeiger auf Wert des String bzw. Stringliteral Syntax: <code>[var Zeiger =] zeiger_auf_string_objekt.valueOf()</code> zeiger_auf_string_objekt auch direkt kodiertes String-Literal
	String()	liefert Zeiger auf String-Wert eines Objektes ist identisch mit Methode .toString() Syntax: <code>[var Kette =] String(Zeiger)</code> Zeiger auch auf Date-Objekt Kette wenn Zeiger auf Date-Objekt, so Datum als String geliefert (je nach Ländereinstellung des Windows) Beispiel: Thu Aug 18 04:37:43 GMT-0700 (Pacific Daylight Time) 1983
Methoden von IE und NS:		
.anchor()	HTML-Anker <A> als HTML-Kette erzeugen in der Form " <code>text</code> " und ohne weitere Attribute ID, HREF etc.	
Beispiel:	<code>var StringLiteral = "Sichtbarer Ankertext"; var HTML_Kette = StringLiteral.anchor("WertDesNAMEAttributes"); HTML_Kette = "Sichtbarer Ankertext".anchor("WertDesNAMEAttributes");</code> entspricht <code>Sichtbarer Ankertext</code> <code>eval(HTML_Kette);</code> erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann <code>document.write(HTML_Kette);</code> zeigt den Anker an und erzeugt Eintrag in der Collection <code>document.anchors</code>	
.big()	HTML-Tag <BIG> erzeugen in der Form <code><BIG>text</BIG></code>	



Beispiel:

```
var StringLiteral    ="Text der BIG wird"
var HTML_Kette      = StringLiteral.big();
HTML_Kette          = "Text der BIG wird".big();
```

entspricht <BIG>Text der BIG wird</BIG>

```
eval(HTML_Kette);          erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

.blink()

HTML-Tag <BLINK> erzeugen in der Form <BLINK>text</BLINK>

Beispiel:

```
var StringLiteral    ="Text der BLINK wird"
var HTML_Kette      = StringLiteral.blink();
HTML_Kette          = "Text der BLINK wird".blink();
```

entspricht <BLINK>Text der BLINK wird</BLINK>

```
eval(HTML_Kette);          erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

.bold()

HTML-Tag erzeugen in der Form text

Beispiel:

```
var StringLiteral    ="Text der BOLD wird"
var HTML_Kette      = StringLiteral.bold();
HTML_Kette          = "Text der BOLD wird".bold();
```

entspricht Text der BOLD wird

```
eval(HTML_Kette);          erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

.charAt()

Zeichen liefern unter Nutzung des Indexes

Beispiel 1:

```
var StringLiteral ="Text"
StringLiteral.charAt(0)    liefert "T"
StringLiteral.charAt(5)    liefert Leerkette
"Text".charAt(0)           liefert "T"
```

Beispiel 2 Zeichenkette auf Wertebereich der Zeichen prüfen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function pruefe_zeichenkette(zeichenkette, wertebereich)
{
    // wertebereich ist Zeichenkette z.B. "0123456789 -+/,()")

    var rueckgabewert = true;          // Annahme: Zeichenkette hat NUR Zeichen
                                        // aus dem Wertebereich

    var zeichen_aus_zeichenkette;

    // Zeichenkette zeichenweise analysieren: Jedes Zeichen mit Wertebereich vergleichen
    for (var i = 0; i < zeichenkette.length; i++)
    {
        zeichen_aus_zeichenkette = zeichenkette.charAt(i);

        if (wertebereich.indexOf(zeichen_aus_zeichenkette) == -1)
        {
            rueckgabewert = false;
            break;    // ungültiges Zeichen gefunden, also abbrechen
        }
    }

    return rueckgabewert;
}

function pruefe_eingabe(zu_pruefende_zeichenkette)
{
    if (pruefe_zeichenkette(zu_pruefende_zeichenkette, "0123456789 -+/,()") )
    {alert("Eingabe ist korrekt !");}
```



```

        else
        {alert("Eingabe ist nicht korrekt !"); }
    }

    //-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>

    Telefon:
    <INPUT  TYPE="text"
           NAME="Telefon"
           VALUE=""

    >
    <INPUT  TYPE="button"
           VALUE="Ueberpruefen"
           onclick="pruefe_eingabe(this.form.Telefon.value)">

</FORM>
</BODY>
</HTML>

```

.charCodeAt() Unicode des Zeichen liefern unter Nutzung des Indexes
Unicode von 0 bis 65535, wobei
0 bis 127 der ASCII-Zeichensatz ist
0 bis 255 der ISO-Latin-1-Zeichensatz ist

Beispiel:

```

var StringLiteral ="Text";
StringLiteral.charCodeAt(0)      liefert 84 für "T"
StringLiteral.charCodeAt(5)      liefert NaN
"Text".charCodeAt(0)             liefert 84 für "T"

```

.concat() Verkettung von String-Objekten bzw. String-Literalen zu einem neuen String-Objekt
Alternative: + Operator der Stringverkettung

Beispiel:

```

var StringLiteral1 ="Hallo ";
var StringLiteral2 ="Du ";
var StringLiteral3 ="!";
var Kette =StringLiteral1.concat(StringLiteral2,StringLiteral3) // "Hallo Du !"
var Kette ="Hallo ".concat("Du ", "!") // "Hallo Du !"

```

.fixed() HTML-Tag <TT> erzeugen in der Form <TT>text</TT>

Beispiel:

```

var StringLiteral    ="Text der TT wird"
var HTML_Kette       = StringLiteral.fixed();
HTML_Kette           = "Text der TT wird".fixed();

```

entspricht <TT>Text der TT wird</TT>

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.fontcolor() HTML-FONT als HTML-Kette erzeugen in der Form "text
und ohne weitere Attribute ID etc.

Beispiel:

```

var StringLiteral    ="Sichtbarer Text"
var HTML_Kette       = StringLiteral.fontcolor("WertDesCOLORAttributes");
HTML_Kette           = "Sichtbarer Text".fontcolor("WertDesCOLORAttributes");

```

entspricht Sichtbarer Text

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.fontsize() HTML-FONT als HTML-Kette erzeugen in der Form "text
und ohne weitere Attribute ID etc.

Beispiel:

```

var StringLiteral    ="Sichtbarer Text"
var HTML_Kette       = StringLiteral.fontsize("WertDesSIZEAttributes");
HTML_Kette           = "Sichtbarer Text".fontsize("WertDesSIZEAttributes");

```

entspricht Sichtbarer Text



eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
 document.write(HTML_Kette);

.fromCharCode() String-Literal erzeugen aus Folge von Unicoden
 Unicode von 0 bis 65535, wobei
 0 bis 127 der ASCII-Zeichensatz ist
 0 bis 255 der ISO-Latin-1-Zeichensatz ist

Beispiel:
 String.fromCharCode(65,66,67) // liefert "ABC"

.indexOf() Suche einer Teilkette in einem String bzw. Stringliteral und die Startposition der Teilkette als Index
 im String bzw. Stringliteral liefern
 es wird nur das ERSTE Auffinden der Teilkette geliefert
 Suche im String erfolgt von links nach rechts
 unterscheidet Gross und Klein

Beispiel 1:

```
var StringLiteral ="StringLiteral";
StringLiteral.indexOf("gLi", 2) liefert 5
StringLiteral.indexOf("gLi") liefert 5
StringLiteral.indexOf("gLi", 10) liefert -1
"StringLiteral".indexOf("gLi", 2) liefert 5
```

Beispiel 2 E-Mail-Adresse auf "@" prüfen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function pruefe_zeichenkette(zeichenkette)
    {
        if (zeichenkette.indexOf('@') = -1)
        {alert("@ fehlt !");}
        else
        {alert("@ gefunden !");}
    }
//-->
</SCRIPT>
</HEAD>

<BODY>
<FORM>
    e-Mail:
    <INPUT TYPE="text"
        NAME="Mail"
        VALUE=""
    >
    <INPUT TYPE="button"
        VALUE="Ueberpruefen"
        onclick=" pruefe_zeichenkette (this.form.Mail.value)"
    >
</FORM>
</BODY>
</HTML>
```

.italics() HTML-Tag <I> erzeugen in der Form <I>text</I>

Beispiel:

```
var StringLiteral ="Text der I wird"
var HTML_Kette = StringLiteral.italics();
HTML_Kette = "Text der I wird".italics();

entspricht <I>Text der I wird</I>
```

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
 document.write(HTML_Kette);

.lastIndexOf() Suche einer Teilkette in einem String bzw. Stringliteral und die Startposition der Teilkette als Index
 im String bzw. Stringliteral liefern
 es wird nur das LETZTE Auffinden der Teilkette geliefert
 Suche im String erfolgt von links nach rechts
 unterscheidet Gross und Klein

Beispiel:



```

var StringLiteral="StringLiteral";
StringLiteral.lastIndexOf("t", 2)      liefert 8
StringLiteral.lastIndexOf("t")         liefert 8
StringLiteral.lastIndexOf("t", 10)     liefert -1
"StringLiteral".lastIndexOf("t", 2)    liefert 8

```

.link() HTML-Link <A> als HTML-Kette erzeugen in der Form "text" und ohne weitere Attribute ID, HREF etc.

Beispiel:

```

var StringLiteral    ="Sichtbarer Ankertext"
var HTML_Kette       = StringLiteral.link("WertDesHREFAttributes");
HTML_Kette           = "Sichtbarer Ankertext".link("WertDesHREFAttributes");

```

entspricht Sichtbarer Ankertext

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette); zeigt den Anker an und erzeugt Eintrag in der Collection document.anchors

.match() Suche in einem String oder String-literal per RegExp-Objekt (siehe dort Methode .exec())

Beispiel 1:

```

var Kette           = "The rain in Spain falls mainly in the plain";
var RegExpAusdruck  = /ain/i;
var Feld1           = Kette.match(RegExpAusdruck);
var Feld2           = "The rain in Spain falls mainly in the plain".match(RegExpAusdruck);

```

Beispiel 2:

```

var zu_durchsuchende_kette = "Otto";
var such_muster             = /(\wOtto)/g; // vom Typ RegExp, also regulärer Ausdruck
var ergebnis_feld         = zu_durchsuchende_kette.match(such_muster);

```

.replace() Suche in einem String oder String-literal per RegExp Objekt (siehe dort Methode .exec()) und gefundenen String ersetzen

Beispiel 1:

```

var Kette           =
    "The man hit the ball with the bat.\nwhile the fielder caught the ball with the glove.";
var RegExpAusdruck  = /The/g;
var Feld1           = Kette.replace(RegExpAusdruck, "A"); // "A" ersetzt "The"
var Feld2           =
    "The man hit the ball with the bat.while the fielder caught the ball with the glove.".replace(
        RegExpAusdruck, "A"); // "A" ersetzt "The"

```

Beispiel 2:

```

function F_ahrenheitTauschenGegen_C_eslius(Kette)
{
    var RegExpAusdruck = /(\d+(\.\d*)?)F\b/g;

    return ( Kette.replace(
        RegExpAusdruck,
        function($0,$1,$2) {return(((1-32) * 5/9) + "C");}
    ) );
}
document.write(F_ahrenheitTauschenGegen_C_eslius ("Wasser kristallisiert bei 32F und siedet bei 212F."));

```

Beispiel 3:

```

var zu_durchsuchende_kette = "Otto Waalkes"; // oder RegExp.input="Otto Waalkes"
var such_muster             = /(\wOtto)/g; // such_muster ist ein regulärer Ausdruck
                                // (Objekt RegExp)
                                // Suchmuster ist Otto, wobei Otto gefunden
                                // werden muss für eine erfolgreiche Suche
                                // anstelle von " ist / zu kodieren
                                // alternativ nicht möglich
                                //      RegExp.input="Otto"
                                //      dann kein detailliertes Suchmuster
                                // Option g alternativ kodierbar per
                                //      RegExp.multiline=true;
var ergebnis_kette         = zu_durchsuchende_kette.replace(such_muster, "Heinrich");

```

.search() Suche in einem String oder String-literal per RegExp-Objekt (siehe dort Methode .exec())

Beispiel 1:



```

var Kette           = "The rain in Spain falls mainly in the plain.";
var RegExpAusdruck  = /falls/i;
var Wert            = Kette.search(RegExpAusdruck);

```

Beispiel 2:

```

var zu_durchsuchende_kette = "Otto";
var such_muster            = /(wOtto)/g; // vom Typ RegExp, also regulärer Ausdruck
var ergebnis              = zu_durchsuchende_kette.search(such_muster);

```

.slice() Teilkette aus String oder Stringliteral liefern als neuen String

Beispiel:

```

var StringLiteral = "StringLiteral";
StringLiteral.length liefert 13
StringLiteral.slice(3,-5) liefert "ingLit" // 13 + (-5) ergibt 8
StringLiteral.slice(3,5) liefert "ing"
"StringLiteral".slice(3,5) liefert "ing"

```

.small() HTML-Tag <SMALL> erzeugen in der Form <SMALL>text</SMALL>

Beispiel:

```

var StringLiteral = "Text der SMALL wird"
var HTML_Kette    = StringLiteral.small();
HTML_Kette        = "Text der SMALL wird".small();

```

entspricht <SMALL>Text der SMALL wird</SMALL>

```

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

```

.split() String bzw. Stringliteral zerlegen und als Feld der Teilketten liefern
Zerlegung von links nach rechts
Feldelemente-Folge wie Zerlegungsfolge

Beispiel:

```

var StringLiteral = "StringLiteral";
var Feld = StringLiteral.split("r", 2); Feld enthält 2 Elemente mit "St" und "ingLiteral"
Feld = StringLiteral.split("r", 1); Feld enthält 1 Elemente mit "St"
Feld = StringLiteral.split("r"); Feld enthält 3 Elemente mit "St" und "ingLite" und "al"
Feld = "StringLiteral".split("r"); Feld enthält 3 Elemente mit "St" und "ingLite" und "al"

```

.strike() HTML-Tag <STRIKE> erzeugen in der Form <STRIKE>text</STRIKE>

Beispiel:

```

var StringLiteral = "Text der STRIKE wird"
var HTML_Kette    = StringLiteral.strike();
HTML_Kette        = "Text der STRIKE wird".strike();

```

entspricht <STRIKE>Text der STRIKE wird</STRIKE>

```

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

```

.sub() HTML-Tag <SUB> erzeugen in der Form _{text}

Beispiel:

```

var StringLiteral = "Text der SUB wird"
var HTML_Kette    = StringLiteral.sub();
HTML_Kette        = "Text der SUB wird".sub();

```

entspricht <SUB>Text der SUB wird</SUB>

```

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

```

.substr() Teilkette aus String oder Stringliteral liefern als neuen String

.substring() Teilkette aus String oder Stringliteral liefern als neuen String

Beispiel:

```

var StringLiteral = "StringLiteral";
StringLiteral.substring(3,3) liefert ""
StringLiteral.substring(3,5) liefert "in"
"StringLiteral".substring(3,5) liefert "in"

```

.sup() HTML-Tag <SUP> erzeugen in der Form ^{text}

Beispiel:



```
var StringLiteral    ="Text der SUP wird"
var HTML_Kette      = StringLiteral.sup();
HTML_Kette          = "Text der SUP wird".sup();
```

entspricht ^{Text der SUB wird}

```
eval(HTML_Kette);           erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

`.toLowerCase()` String bzw. Stringliteral nach neuen String kopieren und dort nach Kleinbuchstaben umwandeln

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.toLowerCase()    liefert "stringliteral"
"StringLiteral".toLowerCase()  liefert "stringliteral"
```

`.toUpperCase()` String bzw. Stringliteral nach neuen String kopieren und dort nach Grossbuchstaben umwandeln

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.toUpperCase()    liefert "STRINGLITERAL"
"StringLiteral".toUpperCase()  liefert "STRINGLITERAL"
```

zusätzliche Methoden vom IE:

`.localeCompare()` Vergleich zweier Strings oder Stringliterals bezüglich ihrer Sortierfolge laut aktuelle Vorgaben zur Sortierungsfolge auf dem PC des Users
nur IE ab 5.5

`.toLocaleLowerCase()` String bzw. Stringliteral nach neuen String kopieren und dort nach Kleinbuchstaben umwandeln
laut aktuelle Sprach-Einstellungen der Umgebung auf PC des Users
nur IE ab 5.5
siehe Script-Objekt String

`.toLocaleUpperCase()` String bzw. Stringliteral nach neuen String kopieren und dort nach Grossbuchstaben umwandeln
laut aktuelle Sprach-Einstellungen der Umgebung auf PC des Users
nur IE ab 5.5
siehe Script-Objekt String

4.3. vordefinierte Objekte zum Browser (Auswahl)

4.3.1. Ansatz

4.3.1.1. vordefinierte Objekte in Javascript /JScript

Es können alle Objekte aus Javascript/JScript mit deren Eigenschaften und Methoden in sinnvoller Kombination mit den zum Browser vordefinierten Objekten verwendet werden. Die objekt-übergreifenden, also objekt-unabhängigen Methoden, sind alle nutzbar aber z.T. in den zum Browser vordefinierten Objekten abgewandelt implementiert.

4.3.1.2. Browserfenster und HTML-Dokument (Objekt window und Objekt document)

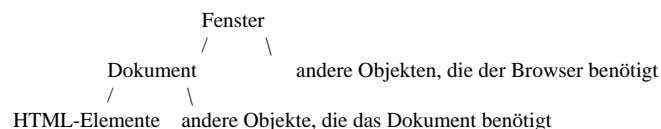
Ein HTML-Dokument muss im Browserfenster nicht angezeigt werden, wenn das HTML-Element keine sichtbaren HTML-Elemente besitzt. So gesehen kann also ein HTML-Dokument nichts mit dem Browserfenster zu tun haben.

Der Browser kann allerdings nur dann ein HTML-Dokument verwalten, wenn es einem Browserfenster zugeordnet ist. So gesehen muss also ein HTML-Dokument immer in einem Fenster liegen, egal ob das Dokument sichtbar ist oder nicht.

Ein Browserfenster verwaltet das HTML-Dokument als Ganzheit (Container)
verwaltet **nicht** die HTML-Elemente des Dokumentes
visualisiert das HTML-Dokument
besitzt weitere Objekte für die Realisierung eines Fensters

Das HTML-Dokument verwaltet seine Elemente anhand des HTML-DOM (siehe weiter unten)
besitzt weitere Objekte und andere Elemente für die Verwaltung

Aus Sicht der gesamten Browserverwaltung wird folgende Hierarchie gebildet:



Zur Umsetzung der Hierarchie werden Zeiger verwendet:

Zeiger als Eigenschaften des Fensters (Objekt window)

Zeiger auf das HTML-Dokument



Zeiger auf andere benötigte Objekte

Zeiger als Eigenschaften des HTML-Dokumentes (Objekt window.document)

Zeiger auf die HTML-Elemente
Zeiger auf andere benötigte Objekte

Damit gilt: Jedes Fenster hat **sein** HTML-Dokument wie umgekehrt.

Solange der Scriptcode für das aktuelle Fenster gilt, muss der Zeiger auf das aktuelle Fenster (Zeiger ist das Schlüsselwort window) nicht kodiert werden.

Beispiel: window.document. ist synonym zu document.

In allen anderen Fällen muss die Punktnotation in der Form zeiger_auf_das_fenster. verwendet werden.

Beispiel: zeiger_auf_fenster.document.

wobei zeiger_auf_fenster

laut Methode .open() zum Fenster (Objekt window)

oder laut Eigenschaft .opener des Objektes window

(Zeiger auf Elternfenster, das **open()** enthält und das Kind-Fenster öffnet, welches in dieser .opener-Eigenschaft den Zeiger auf das Elternfenster enthält)

oder laut Eigenschaft .parent des Objektes window

(Zeiger auf das **in der Fensterhierarchie übergeordnete** Fenster, das aber nicht das .open() zum Kindfenster enthalten muss; z.B. Zeiger auf das

Fenster, das die FRAMESET-Deklaration enthält, oder das diesem Fenster übergeordnet ist)

Achtung: Das **erste** Browserfenster hat keinen Zeiger, denn es stellt die Browserinstanz dar. Zur Referenzierung dieses Fensters muss in nachfolgenden Fenstern der Fensterhierarchie eine der Eigenschaften

.opener bzw. .parent

oder das Ziel_top

verwendet werden.

Im **ersten** Fenster können in der Fensterhierarchie darunterliegende Fenster referenziert werden. Dazu müssen andere Eigenschaften des Objektes window benutzt werden (siehe dort).

Diese Zeigerzuordnungen spiegelt die Vererbung nur z.T. wider:

Beispiele: Ein Fenster kann nicht an HTML-Elemente vererben
Das HTML-Dokument vererbt z.T. an seine Elemente

Aus Sicht der Vererbung ist die o.g. Hierarchie nicht sehr sinnvoll.

Nachfolgende Beschreibungen

halten sich an diese Hierarchie

zeigen Vererbungen

besitzen in Syntaxbeschreibungen immer Referenzen auf das aktuelle Fenster, so dass window im Regelfall dort nicht kodiert wird.

Beispiel: Beschreibung des HTML-Dokumentes document und nicht window.document, da im aktuellen Fenster window.document und document synonym sind.

4.3.1.3. **HTML-Dokument (Objekt document) und seine HTML-Elemente**

4.3.1.3.1. **HTML-Dokument und die Hierarchie der HTML-Elemente**

HTML-Elemente-Hierarchie im HTML-Quellcode des HTML-Dokumentes

Die Kodierung der HTML-Elemente im HTML-Code der Webseite legt die Folge der HTML-Elemente und damit das Layout der Webseite fest. Der Folge entspricht einer HTML-Elemente-Hierarchie laut Syntaxvorschriften der HTML-Konvention.

HTML-Elemente-Hierarchie der Objekte des HTML-Dokumentes



Mittels Umsetzung des HTML-Dokumentes z.B. zum Zweck der Visualisierung durch die Scriptmaschine wird aus jedem HTML-Element ein Objekt erzeugt, das innerhalb der Objekthierarchie des HTML-Dokumentes automatisch platziert wird. Diese Hierarchie wird als **HTML-Dokument-Objekt-Modell (DOM)** bezeichnet und hängt vom Layout der Webseite sowie von Vorgaben der Scriptmaschine ab.

Effektives Instrument der Abbildung einer Hierarchie ist die Baumdarstellung. Das liegt nahe, denn das HTML-Dokument ist der Container aller anderen HTML-Elemente.

DOM basiert fast ausschließlich auf der Baumstruktur. Die Abzweigungen analog vom Stamm zum Ast nennt man Knoten (Nodes). Jede Abzweigung kann weitere haben. Um diese zu unterscheiden, werden die Begriffe Eltern (Parent) und Kind (Child) verwendet. Ohne die Eltern kann ein Kind nicht existieren. Die Eltern können das Kind prägen. Das Kind kann aber auch von den Eltern differierende Auszeichnungen besitzen.

4.3.1.3.2. HTML-Dokument und HTML-DOM

Das HTML-DOM ist ein **symbolisches** Objekt der gesamten HTML-Seite **mit ihrem Kontext** im Browser. Das symbolische Objekt muss man sich als **Baumhierarchie der HTML-Elemente** im HTML-Dokument vorstellen. Diese Hierarchie ermöglicht es, für alle HTML-Elemente in Form von Objekten **jene gemeinsamen** Eigenschaften und Methoden festzulegen, die für die **Baumverwaltung**, also für die Kommunikation der HTML-Elemente in den Baumhierarchie-Ebenen zuständig sind. Dabei ist zu beachten, dass **nicht alle** Objekte die Baumstruktur verwalten können.

Zugleich ermöglicht DOM, element-spezifische Merkmale (Eigenschaften und Methoden) zu implementieren **und** deren Vererbung an Elemente der tiefer gelegenen Hierarchie-Ebene zu definieren.

Damit stellt das HTML-DOM die Grundsatzlogik für die Programmierung einer HTML-Seite per Javascript/JScript dar. Diese Scriptsprachen sind somit komplett objektorientiert und strukturiert (Punktnotation).

DOM bietet zugleich eine Möglichkeit, die Zeigerverwaltung von Eigenschaften und Methoden für den Programmierer sowie für die Scriptmaschine einfacherer und bezüglich Browserperformance optimaler ablaufen zu lassen. Diese Vereinfachung wurde als **Feldform** für diejenigen Objekte im DOM implementiert, die einen gemeinsamen Kontext, z.B. aufgrund gemeinsamer Eltern im Rahmen der Vererbung, besitzen: Vordefinierte **Collectionen** (Sammlungen) von Zeigern, die sich per **Index** verwalten lassen, vereinfachen den Programmierungsaufwand ungemein. Eine Collection ist also objektübergreifend und damit selbst kein Objekt. Das Prinzip der Collectionsbildung wird übrigens auch für einzelne Objekte realisiert.

Erzeugung von HTML-Elementen per HTML oder Script

HTML-Elemente können per HTML-Tag **oder** per Methode `.createElement()` erzeugt werden. Die Behandlung von per HTML-erzeugten Elementen kann von der per DOM-Methoden erzeugten **abweichen**, da DOM die Art der Erzeugungen **differenziert**. Für HTML-erzeugte Objekte gibt es **immer noch** eigene Methoden (Spezialfälle), obwohl das nicht nötig wäre (vermutlich wegen der schrittweisen Erweiterung des DOM auf alle Elemente im Dokument). Empfehlung: Man verwende - wenn möglich - **nur** Methoden, die HTML- **und** Script-erzeugte Elemente bearbeiten und erreicht dadurch die Vereinheitlichung im Quellcode.

Abbildung von Attributen eines Objektes im DOM

Attribute eines Objektes (egal ob per Script oder HTML erzeugt) sind selbst **Knoten und zugleich Kinder** des Objektes, das die Attribute besitzt. Werte von Attributen sind keine Knoten.

Wirksamkeit des HTML-Elementes

"Sichtbarkeit erst wenn Ende-Tag geparkt wurde" bedeutet: Falls das Objekt ein Ende-Tag hat, kann es sich frühestens nach dem Parsen des Ende-Tags im Layout des Dokumentes auswirken. Eventuell ist ein expliziter Refresh des Dokumentes per explizitem `document.location.reload()` nötig.

Bei Tabellen gibt es ebenfalls die Methode `.refresh()`.

Sichtbarkeit des HTML-Elementes

HTML-Elemente werden **in der Regel** sofort geparkt (vom Browser interpretiert). Dabei ist zu beachten: Die Methoden beeinflussen nicht nur die DOM-Hierarchie sondern auch die Sichtbarkeit im Dokument. Bei Änderung des DOM wird die Sichtbarkeit erst beeinflusst, wenn das Ende-Tag geparkt wurde. Das erfolgt garantiert mit dem Neuladen des Dokumentes per explizitem `document.location.reload()`, da die DOM-Methoden **leider nicht zwingend** das Layout des Dokumentes refreshen.

Änderung des DOM und Quellcode des Dokumentes

Änderungen an der DOM-Hierarchie sind nicht immer kongruent zu der Lage der Objekte im Quellcode des Dokumentes. Man gehe davon aus, dass die Lagen im DOM und im Dokument sich definitiv unterscheiden. Daher ist die Verwendung der Attribute ID bzw. NAME (falls diese zugelassen sind) zwingend nötig, um Eindeutigkeit der Bezüge der Objekte im Quellcode auf resultierende DOM-Objekte zur Laufzeit des Dokumentes herzustellen, wenn mit DOM-Eigenschaften und -Methoden per Script gearbeitet werden soll. Falls eine Collection existiert, kann auch diese verwendet werden.

Konsistenz des DOM und Zeiger

Die DOM-Hierarchie muss zu jedem Zeitpunkt konsistent sein (siehe Methode `.normalize()`). Zeiger dürfen **keinesfalls** ins Leere laufen.



DOM-Elemente entfernen

Das Entfernen eines Objektes aus der DOM-Hierarchie muss **nicht** zwingend mit
der physischen Löschung des Elementes
dem Entfernen von Kindern des Elementes
der geänderten Visualisierung des entfernten Elementes und seiner Kinder
verbunden sein.

Prototyping

Objekte können mit neuen Eigenschaften und Methoden erweitert werden (Prototyping per Eigenschaft .prototype). Um diese aber für den Browser verwaltbar zu machen, müssen diese Eigenschaften/Methoden **die vom Browser** unterstützten Eigenschaften und Methoden **letztendlich** aktivieren. Nur mit letzteren lässt der Browser eine Verarbeitung zu, weil er nur **SEINE** implementierten Eigenschaften und Methoden für das Rendern (Darstellen im Layout) von Objekten kennt.

Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht** die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft .prototype .
Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft .prototype .

DOM und Collectionen

DOM verwaltet eine Reihe von Feldern (Collectionen, Sammlungen) für Zeiger. Diese Felder müssen z.T. für die Parameterversorgung der Methoden - speziell für die Adressierung von Objekten - verwendet werden. Mit diesen Feldern wird DOM flexibler handhabbar. Diese Felder sind keine Objekte, enthalten aber gekapselte Daten des DOM mit genau **definierten** Zugriffsmöglichkeiten und Verwendungszwecken.

Normierung von Bezeichner von Methoden des DOM

Methoden mit dem Wort "Node" im Bezeichner behandeln per HTML und **in der Regel** per Script erzeugte Objekte.

Methoden mit dem Wort "Child" im Bezeichner behandeln Kind-Objekte, die genau 1 Eltern haben müssen.
(Den Begriff "Eltern" gibt es im Deutschen nur in der Mehrzahlform (Pluralform)).

Methoden mit dem Wort "Element" im Bezeichner beziehen sich auf Elemente im DOM, also Objekte,
wobei es Eltern, Kinder, per HTML und **in der Regel** per Script erzeugte Objekte sein können.

Methoden mit dem Wort "Attribut" im Bezeichner verwenden immer das attribute Objekt zum betroffenen Objekt.

Wird im Methodenbezeichner **nicht** zugleich das Wort "Node" verwendet, so werden **in der Regel** HTML-erzeugt Attribute behandelt, also **keine** Attribute, die per Methode .createAttribute() erzeugt wurden.

erfolgtes komplettes Laden Dokumentes mit Parsen von </BODY>

"Nur nach dem kompletten Laden des Dokumentes möglich" bedeutet: Der Browser muss das Ende-Tag des BODY geparkt haben, also </BODY>.

fehlendes BODY-Tag beim Internet Explorer

Wurde BODY nicht kodiert und befinden sich im HEAD-Teil des Dokumentes NICHT-HTML-DOM-konforme Anweisungen, die das Rendern von Objekten (Darstellung im Layout des Dokumentes) bewirken (z.B. document.write() mit Erzeugung eines HTML-Elementes), dann erzeugt der Internet Explorer genau dann **automatisch** ein BODY-Objekt, sobald das **erste** HTML-Element erzeugt wird.

Versionen von Browser, Scriptmaschine, HTML und CSS

Man beachte dabei zwingend die Unterschiede zwischen Browsern verschiedener Hersteller **und** Versionen von Browsern ein und desselben Herstellers, sowie für beide Fälle die Browser-Implementierung (Scriptmaschinen-Implementierungen) im jeweiligen Betriebssystem. Das Beachten ist Sache des Programmierers, welches noch durch die Verschiedenheit der Implementierung von Standards zu HTML-Versionen und bezüglich DHTML zu CSS- und Script-Versionen (inklusive deren Umfang und browserhersteller-spezifischen Erweiterungen) erschwert wird. Man gehe davon aus, das Web-Projekte laufend zu pflegen sind, vor allem wenn DHTML zum Einsatz kommen soll (Kombination von HTML, Style-Sheets (CSS) und Scripten), oder man schränke sich auf weitverbreitete Browsertypen und -versionen ein (Verlust von Web-Seiten-Besuchern, die nicht-typische Browser benutzen) bzw. programmiere pures HTML mit Elementen, die von allen Browsern unterstützt werden.

Verbreitete Web-Editoren unterstützen in der Regel nicht das komplette Spektrum der Browserfähigkeiten, sondern decken typische und bedarfsgerechte Fähigkeiten ab. Das komplette Browser-Spektrum lässt sich z.Z. nur durch Programmierung von Hand unter Beachtung des DOM nutzen.

Abwärtskompatibilität des DOM und deprecated Elemente in DOM und HTML



Die **nicht** abwärtskompatible Änderung des DOM durch den Browserhersteller bzw. die zwischenzeitliche Aufgabe von HTML-Tags etc. (deprecated Elemente) ist ebenfalls zu beachten. Diese veralteten Elemente werden zwar z.T. noch unterstützt, sollten aber dringendst nicht mehr verwendet werden. Es ist Sache des Programmierers, diesbezüglich auf dem Laufenden zu sein.

Microsoft JScript.NET als Nachfolger von Microsoft JScript

Bezüglich Script favorisiert Microsoft inzwischen JScript.NET, welches abwärtskompatibel zu JScript ist, aber mehr in die Richtung der Sprachen Java oder C geht (z.B. lassen sich jetzt eine Objektklasse als Prototyp definieren und ein Datentyp bei der Variablendeklaration vergeben). Damit ist auch klar, dass sich DOM geändert hat.

Im Rahmen der NET-Produktpalette für Windows ab NT 5.x (Windows XP) wurde auch der Sprachumfang von Visual Basic verändert, allerdings in massiverem Umfang als zu JScript, so dass für VB-Programmierer ein Umdenken erfolgen **muss**.

XHTML als Nachfolger von HTML4.x

XHTML als "Nachfolger" ist eigentlich eine Erweiterung zu HTML 4.x per XML. XML wird in dieser Dokumentation nicht behandelt.

Microsoft hat XML und HTML ab IE 5.x verbunden bezüglich HTML-DOM und XML-DOM. Z.B. lassen sich XML-Dateninseln inmitten eines HTML-Dokumentes integrieren: Eine im Layout fertige Tabelle wird mit Daten gefüllt, ohne dass diese in das Tabellenlayout eingetragen werden müssen (nur die XML-Platzhalter für Daten). Vorteilhaft wird das dann erst richtig, wenn eine Datenbank die Datensätze in XML-Form erzeugen kann. Anstelle der XML-Dateninseln wird im HTML-Dokument und dessen Tabelle ein Bezug auf die XML-Dateien mit den Datensätzen erzeugt und die Tabelle kann dann diese Datensätze anzeigen. (Selbstverständlich müssen anzuzeigende Daten alle dem jeweiligen Datentyp der im Tabellenlayout kodierten XML-Platzhalter entsprechen). Das Tabellenlayout kann weiterhin mit CSS und Javascript, z.B. JScript, so dynamisch gehalten werden, dass beliebig viele Datensätze in das Tabellenlayout reinpassen (dynamische Tabellenverwaltung ist mit JScript über das DOM der Tabelle (TOM) programmierbar). Unter welchem Betriebssystem die Datenbank läuft, ist egal, hauptsächlich die Datenbank kann XML-konforme Datensätze als Ausgabe erzeugen. Damit wird HTML zur Anzeigeschnittstelle für beliebige Datenbanken. Der Pferdefuss: XML wird objektorientiert realisiert. Damit muss Javascript, z.B. JScript, verwendet werden, mit dem die Zeiger verwaltet werden können (zumal das Tabellenlayout zugleich auch dynamisch gehalten werden muss).

XML dient vorrangig als Datenbeschaffung z.B. für Webseiten mit dynamischen Inhalt im vorgefertigtem dynamischen Layout. Unter der NET-Software von Windows XP wird XML netzwerkfähig und hat zusätzliche Layoutkomponenten (erweitertes XML-DOM), die aber mithin starr bleiben, so dass weiterhin JScript.NET und CSS verwendet werden sollte.

XML wird weiterhin als Schnittstelle für die Datenübergabe zwischen diversen Datenbanken verwendet, die ihre Daten nur im XML-Format erzeugen bzw. einlesen müssen. XML ist es völlig egal, wie die Daten in der Datenbank selbst verwaltet werden. So gesehen wird XML andere Schnittstellen wie IDE ersetzen und z.B. eine HTML-orientierte Datenanzeige per Webserver im Intranet eines Unternehmens oder im Internet (e-commerce) begünstigen. Wer XML-Software entwickelt, kann auch auf einfachere Weise HTML-bezogene Datenbankenausgaben erzeugen, als es z.B. das Unternehmen SAP mit ABAP4-eigenen HTML-Erstellungsroutinen realisiert, die nicht XML nutzen und die daher auch nur von SAP-Software bzw. für SAP programmierte Software beherrscht werden. Zugleich sind andere softwarebedingte Ausgabeformate wie RTF ersetzbar, da Layoutkomponenten in XML genormt sind, ausschliesslich in Script-Form vorliegen und maschinell sowie vom Menschen lesbar sind. Mit anderen Worten: Ob sich Datenbankhersteller und Nutzer mit herstellereinspezifischer Software für Datenanzeige herumschlagen oder ob beide sich auf XML einigen werden: Es ist eine Frage der Kosten, Konsistenz, Transparenz und vereinfachten Normung der netzwerkbezogenen Datenbank-Datenbereitstellung an den Endverbraucher, zu dessen Zweck die Daten ja verwaltet werden und damit dem Verbraucher Kosten verursachen.

Wer als privater Programmierer Datenbanken nutzt und Daten dynamisch, aber in ihrer Form und im Anzeige-Layout genormt darstellen will, sollte sich mit XML und JScript beschäftigen. Wird z.B. unter Linux die Datenbank gehalten, aber für Windows das Webdesign erstellt, müssen eine Datendateischnittstelle zwischen Linux und Windows geschaffen und die Datenbank zur XML-konformen Datensatzausgabe überredet werden. Anschliessend werden die XML-Dateien unter Windows im HTML-Dokument **ingelesen** und angezeigt. Scriptsprachen, die HTML- und XML-Code nicht verwalten bzw. erzeugen können, bleiben allerdings außen vor.

Plain-Text, HTML-Tags und Scriptcode

Für Text-Objekte sind Plain-Text (ohne HTML-Tags und Script) und Text mit HTML-Tags und/oder Scriptcode zu unterscheiden. Besonders spielt das eine Rolle bei der Programmierung des DIV-Objektes.

wichtige Hinweise:

Hinweis zur Verwendung logischer Objektnamen für HTML-Elemente und browserinterne Objekte:

Durch Referenzierung eines Objektes (implizit per HTML-Tag-Kodierung oder automatisch durch den Browser) wird ein interner Zeiger gebildet, über den das Objekt erreichbar ist. Zusätzlich ist im HTML-Tag eventuell die Verwendung des Attributes ID möglich, um einen logischen Namen zum Objekt zu vergeben. Dieser logische Name wird vom Browser als Referenz auf das interne Objekt erkannt. ID ist also die öffentliche Schnittstelle zum Programmierer, der ansonsten nur schwer an interne Zeiger gelangen kann (spezielle Eigenschaft beim Internet Explorer vorhanden). ID ist also ein Zeiger, der mit seinem Bezeichner zur Referenzierung in Script direkt per Punktnotation kodiert werden kann. So gesehen ist der Zeiger für den Programmierer eine **logische** Größe. Daher wird nachfolgend auch oft beim Wert des ID-Attributes oder NAME-Attributes vom **logischen** Namen des HTML-Elementes gesprochen.

Die Verwendung von NAME bzw. ID ist nicht für jedes HTML-Objekt möglich. Es ist also die HTML-Beschreibung des Objektes zu beachten. Wie immer ist zu prüfen, ob der Browserhersteller überhaupt das Tag unterstützt und wenn ja, in welcher Browserversion mit welchen Attributen, also z.B. mit oder ohne NAME bzw. ID, und ob diese Attribute auch in JavaScript unterstützt werden. Warum eine



einheitliche Schnittstelle über NAME bzw. ID zu allen HTML-Objekten nicht vorhanden ist, bleibt unklar wie der Fakt, dass HTML trotz existierendem HTML-Standard verschiedenartig in den Browsern implementiert ist.

Ist ein HTML-Objekt im Browser implementiert, so existiert auch eine interne objektorientierte Zeigerverwaltung, aber nicht zwingenderweise **die** Schnittstelle zum Programmierer. Daher ist es verständlich, wenn Java-Script-Programmierer für eine dynamische HTML-Programmierung (DHTML) denjenigen Browser vorziehen, der ein Maximum an Schnittstellen bietet. Diese Bevorzugung steht im Widerspruch zu den Gepflogenheiten der Browsernutzer, abgesehen von den Marktpositionen, der Schnelligkeit und eventuellen Sicherheitsproblemen der Browser.

Der logische Name kann für Zugriffe auf das Objekt verwendet werden. Die Art des Zugriffs ist den nachfolgenden Objekt-Beschreibungen zu entnehmen. Aber gerade bei Zugriffen über eine Objekteigenschaft als Feld ist der logische Name sehr sinnvoll: Er erspart die Indexermittlung.

Bsp:

```
<DIV ID="Logischer_DIV_name" .....> ...>/DIV>
```

```
document.all.Logischer_DIV_name.style.left=20; // Internet Explorer
document.layers[Logischer_DIV_name].left=20; // Netscape unter 6.x
```

In nachfolgenden Objektbeschreibungen wird bei Feldern immer der Zugriff per Index angegeben. Daher ist für diese Fälle nicht zu vergessen, dass ein vorhandener logischer Name anstelle des Indexes verwendet werden kann. Der Zugriff über den Index funktioniert immer. Der Zugriff über den logischen Namen kann nur funktionieren, wenn das HTML-Objekt diesen als kodierbar zulässt und der Browser diese Schnittstelle unterstützt.

Hinweise zur Beschreibung von Browserobjekten:

Betriebssystem-nahe Objekte werden z.T. **nicht** beschrieben.

Es wird das Betriebssystem MS Windows 32-Bit unterstellt – ohne Differenzierung nach den Varianten.

Schwerpunkt der Beschreibungen bildet der Microsoft Internet Explorer ab 6.x unter Windows 32-Bit.

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die
Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche
Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zur dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert, wenn gleiche Attributnamen bzw. Eigenschaften betroffen sind (sonst hinzufügen).

Es ist zu beachten, dass der Internet Explorer diverse Collectionen hat, die Zeiger von Objekten zur Laufzeit mit den **aktuellen Werten** von Objekt-Eigenschaften verwalten. Diese Collectionen

werden **leider nur z.T.** automatisch aktualisiert
sind anzusprechen, wenn es um laufzeitaktuelle Eigenschaften geht
dienen dem internen Ablauf der Scriptmaschine.

Z.T. werden Objekte zur Laufzeit zugleich **verschieden** verwaltet, auch wenn es sich um die gleiche Eigenschaft handelt. Damit taucht die Objekt-Eigenschaft wertmäßig mehrmals auf - eine gewollte Redundanz des Herstellers der Scriptmaschine, auch um Kompatibilität der Versionen zu erhalten. (Programmierer Code sollte bezüglich Instanzen von Objekten und deren gekapselten Daten sowie bezüglich der Variablenverwaltung im Quellcode möglichst redundanzfrei sein.)

Windows 9x ist kein Echtzeitsystem, so dass sichtbare Eigenschaftenveränderungen **nicht** selbstverständlich **synchron** laufen müssen, auch wenn sie als synchron programmiert wurden. Für Synchronisierung spielt neben der CPU-Geschwindigkeit immer noch das **Multitasking in Echtzeit** eine Rolle. Der Programmierer sollte das testen und sich nicht auf die synchrone Programmierung verlassen (Synchron-Probleme sind für bestimmte Objekte an entsprechender Stelle in dieser Dokumentation kommentiert).

Das Laden von Daten zu einem HTML-Element, z.B. Laden von mehreren (und danach auf 1 Schlag anzuzeigenden) Bildern, erfolgt parallel. Außerdem unterliegt der Internetzugang diversen Schwankungen (auch bei ADSL etc.). Damit sind Synchron-Probleme objektiv möglich. Auch wenn JScript nur z.T. objektspezifische Hilfsmittel bietet (Events), die signalisieren, wann ein Laden beendet ist, werden diese Events ebenfalls parallel erzeugt. Programmtechnisch hieße das für das Laden der Bildfolge: Es muss das "Einsammeln" aller Events in einer Routine programmiert werden, die solange aktiv ist, bis das **letzte** Event gemeldet wurde, was aber leider **nicht** bedeutet, dass die Bildfolge auch mit Eintreten des letzten Events komplett angezeigt ist und schon garnicht alle Bilder auf 1 Schlag. Gewisse Abhilfe schafft nur das Vorladen von Daten, also im Falle der Bilder: In einer Schleife das



Instanzieren eines unsichtbaren IMG-Objektes per new-Anweisung (new Image()) **mit Urlzuweisung** pro Bild **und** das Abfragen der Events auf Ladezustand pro Bild. Ende der Schleife ist das Melden des letzten Events. Erst danach sind **alle** Bilder sichtbar zu machen (style.visibility). Aber auch letzteres muss nicht Synchronisierung der Anzeige erzeugen, kann es aber zumindest besser, als das nicht steuerbare Ladeverhalten von Daten ohne "Einsammel"-Routine. (Wichtig: Bei Bildern immer in der new Image()-Anweisung die korrekte Bilddimension angeben, denn die wird zur Anzeigeberechnung verwendet. Die Bilddimension sollte zur Dimension des Umfeldes (Kontextes) passend sein, z.B. zur Dimension des DIV's, der das Bild umgibt, wenn das Bild später anhand des DIV's z.B. bewegt werden soll).

Hinweise zu Syntaxbeschreibungen:

String bedeutet Zeichenkette oder Zeichenketten-Literal oder numerisches Literal
 Integer bedeutet ganzzahlig (32-Bit), positiv, negativ, inklusive 0
 Floating point bedeutet Kommazahl mit Punkt als Komma, positiv, negativ, inklusive 0

Die Symbole [] schließen optionale Größen ein, die also nicht kodiert werden müssen, aber können. wobei z.T. Verschachtelung möglich ist

Bsp.: [parameter1 [,parameter2 [,parameter3]]]

Parameter 1 bis 3 sind optional
 Parameter 2 bis 3 sind optional
 Parameter 3 ist optional
 mindestens Parameter 1 kodierbar
 für Parameter3 muss zuvor Parameter2 kodiert werden
 wenn Parameter 2 und / oder Parameter 3 kodiert, so mit Komma

Achtung: Bei Feldern und Collectionen bedeuten [] die Indexbegrenzer und nicht "optional".

Der Index einer Collection sollte in [] kodiert werden, kann aber auch in () **kodiert sein**, wenn es der Browser zulässt. In Syntaxbeschreibungen von Collectionen sind also die Klammern [] **zum Index nicht** die Zeichen für **optional**. Der Index ist in der Regel Integer ab 0, kann aber eventuell der **logische Objektname** laut Wert des ID-Attributes oder des NAME-Attributes sein. Wenn der logische Objektname zulässig ist, so ist es immer ein String, der in " " bzw. ' ' kodiert sein **kann**, aber nicht muss. Letzteres gilt auch für die Kodierung des Wertes des ID- bzw. NAME-Attributes in einem HTML-Tag.

Instanz bedeutet ein zur Ausführungszeit (RunTime) des Dokumentes existierendes Objekt im Hauptspeicher.

Referenz bedeutet Zeiger, also ein Bezug auf eine Instanz .

Ein Bezug ist die Adresse der Instanz, wobei die Adresse in einem Zeiger gespeichert ist. Für den Programmierer sind also Zeiger und Adresse synonym.

Eine Adresse ist die Position der Instanz im Hauptspeicher. Die Adresse liegt in einer Variablen als Zeiger auf diese Adresse. Hinweis: Die Variable selbst muss ebenfalls eine Adresse haben, worum sich der Programmierer nicht zu kümmern braucht.

Ein Zeiger ist eine Variable, die eine Adresse enthält UND zugleich mit dem Typ der Variablen bekannt gibt, wie der Browser den Wert der Variablen zu verarbeiten hat.

Bsp: var Text = "Otto";

Hinter Text steckt die Adresse der Zeichenkette "Otto" im Hauptspeicher.
 Zeichenkette "Otto" ist der Wert der Variablen Text und liegt an der Position im Hauptspeicher, auf die die Variable zeigt.
 Anhand von "" weiß der Browser, dass es sich um eine Zeichenkette handelt, mit der z.B. nicht addiert werden kann.
 Die Variable Text hat ebenfalls eine Adresse, um die sich der Programmierer nicht zu kümmern braucht.

Objekt oder Element oder HTML-Tag oder HTML-Element sind synonym, wenn es um die objektorientierte Darstellung geht. Ein HTML-Tag ist ein Element eines Dokumentes und zugleich als Objekt realisiert, genau wie das Dokument selbst.

Dokument ist synonym zu HTML-Dokument, das z.B. die Elemente HEAD und BODY hat. Alle 3 sind Objekte !

Die Objektstrukturen im Browser basieren nur zum Teil auf der Kodierungsreihenfolge im Quellcode des Dokumentes. Der Quellcode wird vom Browser interpretiert und ausgeführt, wobei dazu das DOM-Modell des jeweiligen Browsers herangezogen wird, das die Kodierungs-Reihenfolge der Elemente im Quellcode beim Ausführen des Codes verändern kann. Maßgebend ist die Implementierung des DOM in den Browser als Verhaltensregeln für den Browser zur Verarbeitung des Quellcodes und dessen objektorientierte Umsetzung. Es gibt im Prinzip keine Elemente, die nicht Objekt bzw. Teil eines Objektes sind.

Eine Collection ist eine Sammlung von (nicht unbedingt typengleichen) Elementen in Form eines Feldes (array), z.B. eine Sammlung von Zeigern. Eine DOM-Collection dient zur Erweiterung der Baumstruktur anhand von Zeigern als Möglichkeit der direkten Referenzierung, ohne die Baumstruktur ablaufen zu müssen (Zwischenspeicher von Zeigern). Zugriff auf Elemente der Collection erfolgt per Index oder z.T. auch per logischen Name z.B. laut ID-Attribut.

null ist der NULL-Zeiger, der einen Dummy-Wert hat (nicht numerisch Null !!!), also auf nichts weist und nichts referenziert. Eine instanzierte Variable, die den Wert null zugewiesen bekommt, zeigt dann auf nichts, und das Objekt, auf den die Variable **vor** der Null-



Zuweisung gezeigt hat, wird mit der Null-Zuweisung aus dem Hauptspeicher entfernt. Hinweis: Die Adresse der Variablen selbst wird davon nicht betroffen.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit Kompression genutzt wird z.B. bei onlick-Handler auf IMG klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show()erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).



Der Popublocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popublocker hat ein Popupfenster geblockt. Sie können den Popublocker deaktivieren oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popublocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popublocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popublocker des IE bemerkt aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popublocker verwaltet wird.

Der Popublocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popublocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popublockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popublockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

window.focus();

window.document.focus();

if(document.body!=null)

{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)

{document.body.focus();}

}

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

popupzeiger.show(...);

Hinweis: Der Popubfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus

.setActive() ist Teilmenge von .focus(): nur das aktiv setzen

funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtsstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.

Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.



Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)



verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:•

<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich



Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5



Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

4.3.1.3.3. HTML-DOM

DOM wurde browserspezifisch und zugleich abhängig von der Browserversion bei gleichem Browserhersteller implementiert.

Aus Platzgründen sind z.T. die Beispiele in den Anhang verlegt worden. Viele Objekte haben gemeinsame Methoden und Eigenschaften, so dass diese Art der Platzierung für Übersichtlichkeit sorgt.

4.3.1.3.3.1. HTML-DOM beim Netscape 6.x (Übersicht)

HTML-DOM wird ab NS 6.x um Eigenschaften, Methoden sowie um das Objekt `document.documentElement` erweitert.

4.3.1.3.3.1.1. Methoden vom Objekt `document` im HTML-DOM des Netscape

Hinweis: Bezeichner von Tags immer in Grossbuchstaben

4.3.1.3.3.1.1.1. `document.getElementById()` des Netscape

Zeiger auf HTML-Element ermitteln: Element **muss** mit **ID-Attribut** versehen sein (optional zusätzlich Attribut **NAME**).

Bsp: `<P ID="P_ID" > ... </P>`

```
var P_Tag_Zeiger = document.getElementById("P_ID");
```

P_Tag_Zeiger dient zur Referenzierung von eigenschaften und Methoden des P-Elementes

Hinweise:

Der Internet Explorer unterstützt z.T. ebenfalls diese Methode, so dass dann diese Methode nicht zur Identifizierung des NS 6.x verwendet werden kann.

4.3.1.3.3.1.1.2. `document.getElementsByTagName()` des Netscape

Zeiger auf Feld aller Objekte im Dokument ermitteln, die einen gemeinsamen Tag besitzen.

Feldeigenschaft <code>.length</code>	Anzahl der Objekte
Feldmethode <code>.item()</code>	Zeiger auf Feldelement anhand Feldindex liefern
Feldindex	Integer, ab 0

Element muss **nicht** das ID-Attribut kodiert haben (kann aber).

Bsp.: `var Feld_aller_P_Tag = document.getElementsByTagName("P");`

Bsp.: `var ZeigerAufBODY = document.getElementsByTagName("BODY").item(0);` // es gibt nur 1 BODY im Dokument

Bsp.: `var myTable = document.createElement("TABLE");`
`// ... diverse TR erzeugen`
`var TR_Feld = tableBody.getElementsByTagName("TR")`

Anzahl der TR laut `TR_Feld.length` ab 1

Feldelement laut `TR_Feld.item(index_ab_0)`

z.B. `lastRow = TR_Feld.item(allRows.length-1);`
`firstRow = TR_Feld.item(0);`

4.3.1.3.3.1.1.3. `document.createElement()` des Netscape

HTML-Element leer erzeugen und Zeiger liefern, aber leider ohne Einbindung des Objektes in das Dokument.

Beispiel für nicht-komplexes Element wie `<P>`:

```
var P_Tag_Zeiger=document.createElement("P");
```

Beispiel für komplexes Element wie `<TABLE>`:

```
// Zeiger auf BODY holen
var ZeigerAufBODY = document.getElementsByTagName("body").item(0);

// Tabelle leer erzeugen mit ID
var ZeigerAufTabelle = document.createElement("TABLE");
ZeigerAufTabelle.id = "ID_Table";
```

```
// Tabellenkörper leer erzeugen
```




```

var ZeigerAufTBODY = document.createElement("TBODY");

// Tabellenkörper zeilenweise füllen (3 Zeilen)
for (var i = 0; i < 3; i++)
{
    // Tabellenzeile leer erzeugen
    var ZeigerAufTR = document.createElement("TR");

    // Tabellenzeile zellenweise füllen (3 Zellen)
    for (var j = 0; j < 3; j++)
    {
        // Tabellenzelle leer erzeugen
        var ZeigerAufTD = document.createElement("TD");

        // Tabellenzelle dimensionieren
        ZeigerAufTD.setAttribute("WIDTH", "50");
        ZeigerAufTD.setAttribute("HEIGHT", "50");

        // Tabellenzelle mit Text füllen
        var textNode = document.createTextNode("Test");
        ZeigerAufTD.appendChild(textNode);

        // Tabellenzelle an die Tabellenzeile anhängen als Kind der Zeile
        ZeigerAufTR.appendChild(ZeigerAufTD);
    }

    // gefüllte Tabellenzeile an den Tabellenkörper anhängen als Kind des Körpers
    ZeigerAufTBODY.appendChild(ZeigerAufTR);
}

// Tabellenkörper an die Tabelle anhängen als Kind der Tabelle
ZeigerAufTabelle.appendChild(ZeigerAufTBODY);

// Tabelle an BODY des Dokumentes anhängen als Kind des Dokumentes
ZeigerAufBODY.appendChild(ZeigerAufTabelle);

```

4.3.1.3.3.1.1.4. **document.createAttribute() des Netscape**

Attribut eines HTML-Elementes ohne Wert erzeugen.

```
var Attribut_Zeiger = document.createAttribute("attribut_bezeichner");
```

Bsp:

```
var myTable = document.createElement("TABLE");
myTable.id = "TableOne";
myTable.border = 1;
```

4.3.1.3.3.1.1.5. **document.setAttribute() des Netscape**

Attribut eines HTML-Elementes mit Wert erzeugen.

```
P_Tag_Zeiger.setAttribute("attribut_bezeichner", wert);
```

wert ja nach Attribut z.B. als String

Beispiel:

```
var myImg = document.createElement("IMG");
myImg.setAttribute("id", "ID_IMG");
myImg.setAttribute("src", "test.gif");
```

4.3.1.3.3.1.1.6. **document.createTextNode() des Netscape**

Textknoten erzeugen (kein HTML-Knoten)

```
Bsp: var Text_Zeiger = document.createTextNode("freier_text");
```

Bsp: Füllen einer Tabelle:

```

// Zeiger auf BODY holen
var ZeigerAufBODY = document.getElementsByTagName("body").item(0);

// Tabelle leer erzeugen mit ID
var ZeigerAufTabelle = document.createElement("TABLE");
ZeigerAufTabelle.id = "ID_Table";

// Tabellenkörper leer erzeugen
var ZeigerAufTBODY = document.createElement("TBODY");

```



```

// Tabellenkörper zeilenweise füllen (3 Zeilen)
for (var i = 0; i < 3; i++)
{
    // Tabellenzeile leer erzeugen
    var ZeigerAufTR = document.createElement("TR");

    // Tabellenzeile zellenweise füllen (3 Zellen)
    for (var j = 0; j < 3; j++)
    {
        // Tabellenzelle leer erzeugen
        var ZeigerAufTD = document.createElement("TD");

        // Tabellenzelle dimensionieren
        ZeigerAufTD.setAttribute("WIDTH", "50");
        ZeigerAufTD.setAttribute("HEIGHT", "50");

        // Tabellenzelle mit Text füllen
        var textNode = document.createTextNode("Test");
        ZeigerAufTD.appendChild(textNode);

        // Tabellenzelle an die Tabellenzeile anhängen als Kind der Zeile
        ZeigerAufTR.appendChild(ZeigerAufTD);
    }

    // gefüllte Tabellenzeile an den Tabellenkörper anhängen als Kind des Körpers
    ZeigerAufTBODY.appendChild(ZeigerAufTR);
}

// Tabellenkörper an die Tabelle anhängen als Kind der Tabelle
ZeigerAufTabelle.appendChild(ZeigerAufTBODY);

// Tabelle an BODY des Dokumentes anhängen als Kind des Dokumentes
ZeigerAufBODY.appendChild(ZeigerAufTabelle);

```

4.3.1.3.3.1.1.7. **document.createTextRange() des Netscape**

Textbereich erzeugen

Bsp.: var TextRange=document.body.createTextRange();

4.3.1.3.3.1.2. **Objekt document.documentElement des Netscape**

Das Objekt ist ein symbolisches Objekt und repräsentiert ein HTML-Element.

Die Referenzierung des HTML-Elementes erfolgt über document.getElementById() oder document.getElementsByTagName().

Hinweis: Bezeichner von Attributen oder Tags immer in Kleinbuchstaben

4.3.1.3.3.1.2.1. **Eigenschaften von document.documentElement des Netscape**

.attributes	Zeiger auf Feld der Attribute Bsp: var Feld = zeiger_auf_element.attributes;
.attributes.length	Anzahl der Attribute , ab 1
.childNodes	Zeiger auf Feld der Kinderknoten Bsp: var Feld = zeiger_auf_element.childNodes;
.childNodes.length	Anzahl der Kinder, ab 1 Bsp: var Anzahl_kinder = zeiger_auf_element.childNodes.length;
.childNodes.nodeValue	Zeiger auf Wert eines Kindes Bsp: var Wert_des_kindes = zeiger_auf_element.childNodes.item(index_ab_0).nodeValue;
.firstChild	Zeiger auf das erste Kind für das HTML-Dokument wird immer HEAD geliefert Bsp: var zeiger_auf_erstes_kind = zeiger_auf_element.firstChild;
	Hinweis: document.documentElement.firstChild.ownerDocument.documentElement.nodeName liefert HTML
.lastChild	Zeiger auf das letzte Kind Bsp: var zeiger_auf_letztes_kind = zeiger_auf_element.lastChild;
	Hinweis: document.documentElement.lastChild.ownerDocument.documentElement.nodeName



liefert BODY

.nodeName	Zeiger auf Eigenschaften eines Elementes, egal ob Eltern oder Kind Bsp.: var Element_Tag = zeiger_auf_element.nodeName;
.nodeValue	Zeiger auf Wert eines Knotens Bsp.: var Element_Wert = zeiger_auf_element.nodeValue;
.nodeType	Zeiger auf Knotentyp Bsp.: var Element_Typ = zeiger_auf_element.nodeType; z.B. 1 = Tag wie HTML, BODY HEAD etc. 2 = Attribut wie ID, NAME etc. 3 = reiner Text der Tags wie z.B. in. <A>
.ownerDocument	Zeiger auf das HTML-Dokument Bsp: var zeiger_auf_html_dokument = zeiger_auf_element.ownerDocument;
.parentNode	Zeiger auf Eltern-Element Bsp: var zeiger_auf_eltern = zeiger_auf_element.parentNode;
.text	den Text des HTML-Elementes liefern, das die Eigenschaft .text unterstützt z.B. bei OPTION Bsp: var Text = zeiger_auf_element.text;
.value	den Wert des HTML-Elementes liefern, das die Eigenschaft .value unterstützt z.B. SELECT Bsp: var Wert = zeiger_auf_element.value;

4.3.1.3.3.1.2.2. Methode von document.documentElement des Netscape

.appendChild()	HTML-Element einbinden als Kind Bsp: zeiger_auf_eltern.appendChild(zeiger_auf_zu_neues_kind);
.attributes.item()	Zeiger auf Attribut eines HTML-Elementes ermitteln Bsp: var Attribut_Zeiger=P_Tag_Zeiger.attributes.item(index_des_attributes); alle Attribute werden ab 0 in der Kodierungsrichtung von links nach rechts indiziert 0 ist also das linkeste Attribut, also das erste
.childNodes.item()	Zeiger auf ein Kind Bsp: var zeiger_auf_kind = zeiger_auf_element.childNodes.item(index_ab_0);
getAttribute()	Wert eines Attribut ermitteln Bsp: var wert=P_Tag_Zeiger.getAttribute("attribut_bezeichner");
.hasChildNodes()	prüfen ob Kinder vorhanden sind Bsp: var kinder_vorhanden = zeiger_auf_element.hasChildNodes(); vorhanden so true geliefert
.insertBefore()	HTML-Element einfügen zeiger_auf_eltern.insertBefore(zeiger_auf_einzufügendes_kind, zeiger_auf_kind_vor_dem_eingefügt_wird); Beispiel: // Zeiger auf BODY holen var ZeigerAufBODY = document.getElementsByTagName("body").item(0); // Tabelle leer erzeugen mit ID var ZeigerAufTabelle = document.createElement("TABLE"); ZeigerAufTabelle.id = "ID_Table"; // Tabellenkörper leer erzeugen var ZeigerAufTBODY = document.createElement("TBODY"); // Tabellenkörper zeilenweise füllen (3 Zeilen) for (var i = 0; i < 3; i++) { // Tabellenzeile leer erzeugen



```

var ZeigerAufTR = document.createElement("TR");

// Tabellenzeile zellenweise füllen (3 Zellen)
for (var j = 0; j < 3; j++)
{
    // Tabellenzelle leer erzeugen
    var ZeigerAufTD = document.createElement("TD");

    // Tabellenzelle dimensionieren
    ZeigerAufTD.setAttribute("WIDTH", "50");
    ZeigerAufTD.setAttribute("HEIGHT", "50");

    // Tabellenzelle mit Text füllen
    var textNode = document.createTextNode("Test");
    ZeigerAufTD.appendChild(textNode);

    // Tabellenzelle an die Tabellenzeile anhängen als Kind der Zeile
    ZeigerAufTR.appendChild(ZeigerAufTD);
}

// gefüllte Tabellenzeile an den Tabellenkörper anhängen als Kind des Körpers
ZeigerAufTBODY.appendChild(ZeigerAufTR);
}

// Tabellenkörper an die Tabelle anhängen als Kind der Tabelle
ZeigerAufTabelle.appendChild(ZeigerAufTBODY);

// Tabelle an BODY des Dokumentes anhängen als Kind des Dokumentes
ZeigerAufBODY.appendChild(ZeigerAufTabelle);

var ZeigerAufTRNeu = document.createElement("TR");
var ZeigerAufTRAlt = document.getElementsByTagName("TR").item(0);
ZeigerAufBODY.insertBefore(ZeigerAufTRNeu, ZeigerAufTRAlt);

```

`.removeAttribute()` Attribut eines HTML-Elementes mit Wert löschen.

Bsp: `var geloeschtes_Attribut_Zeiger = P_Tag_Zeiger.removeAttribute("attribut_bezeichner");`

leider ist ein Stringparameter zu übergeben und nicht der Zeiger auf das Attribut.

`.removeChild()` Kind entfernen

Bsp: `zeiger_auf_eltern.removeChild(zeiger_auf_kind);`

Beispiel:

```

// Zeiger auf BODY holen
var ZeigerAufBODY = document.getElementsByTagName("body").item(0);

// Tabelle leer erzeugen mit ID
var ZeigerAufTabelle = document.createElement("TABLE");
ZeigerAufTabelle.id = "ID_Table";

// Tabellenkörper leer erzeugen
var ZeigerAufTBODY = document.createElement("TBODY");

// Tabellenkörper zeilenweise füllen (3 Zeilen)
for (var i = 0; i < 3; i++)
{
    // Tabellenzeile leer erzeugen
    var ZeigerAufTR = document.createElement("TR");

    // Tabellenzeile zellenweise füllen (3 Zellen)
    for (var j = 0; j < 3; j++)
    {
        // Tabellenzelle leer erzeugen
        var ZeigerAufTD = document.createElement("TD");

        // Tabellenzelle dimensionieren
        ZeigerAufTD.setAttribute("WIDTH", "50");
        ZeigerAufTD.setAttribute("HEIGHT", "50");
    }
}

```



```

        // Tabellenzelle mit Text füllen
        var textNode = document.createTextNode("Test");
        ZeigerAufTD.appendChild(textNode);

        // Tabellenzelle an die Tabellenzeile anhängen als Kind der Zeile
        ZeigerAufTR.appendChild(ZeigerAufTD);
    }

    // gefüllte Tabellenzeile an den Tabellenkörper anhängen als Kind des Körpers
    ZeigerAufTBODY.appendChild(ZeigerAufTR);
}

// Tabellenkörper an die Tabelle anhängen als Kind der Tabelle
ZeigerAufTabelle.appendChild(ZeigerAufTBODY);

// Tabelle an BODY des Dokumentes anhängen als Kind des Dokumentes
ZeigerAufBODY.appendChild(ZeigerAufTabelle);

var ZeigerAufTR = document.getElementsByTagName("TR").item(0);
ZeigerAufBODY.removeChild (ZeigerAufTRNeu, ZeigerAufTR);

.replaceChild()      HTML-Element ersetzen
                    Bsp:      zeiger_auf_eltern.replaceChild( zeiger_auf_zu_neues_kind,
                                                                zeiger_auf_zu_ersetzendes_kind
                                                                );

Beispiel:
// Zeiger auf BODY holen
var ZeigerAufBODY = document.getElementsByTagName("body").item(0);

// Tabelle leer erzeugen mit ID
var ZeigerAufTabelle = document.createElement("TABLE");
ZeigerAufTabelle.id = "ID_Table";

// Tabellenkörper leer erzeugen
var ZeigerAufTBODY = document.createElement("TBODY");

// Tabellenkörper zeilenweise füllen (3 Zeilen)
for (var i = 0; i < 3; i++)
{
    // Tabellenzeile leer erzeugen
    var ZeigerAufTR = document.createElement("TR");

    // Tabellenzeile zellenweise füllen (3 Zellen)
    for (var j = 0; j < 3; j++)
    {
        // Tabellenzelle leer erzeugen
        var ZeigerAufTD = document.createElement("TD");

        // Tabellenzelle dimensionieren
        ZeigerAufTD.setAttribute("WIDTH", "50");
        ZeigerAufTD.setAttribute("HEIGHT", "50");

        // Tabellenzelle mit Text füllen
        var textNode = document.createTextNode("Test");
        ZeigerAufTD.appendChild(textNode);

        // Tabellenzelle an die Tabellenzeile anhängen als Kind der Zeile
        ZeigerAufTR.appendChild(ZeigerAufTD);
    }

    // gefüllte Tabellenzeile an den Tabellenkörper anhängen als Kind des Körpers
    ZeigerAufTBODY.appendChild(ZeigerAufTR);
}

// Tabellenkörper an die Tabelle anhängen als Kind der Tabelle
ZeigerAufTabelle.appendChild(ZeigerAufTBODY);

// Tabelle an BODY des Dokumentes anhängen als Kind des Dokumentes
ZeigerAufBODY.appendChild(ZeigerAufTabelle);

var ZeigerAufTRNeu = document.createElement("TR");

```



```
var ZeigerAufTRAlt = document.getElementsByTagName("TR").item(0);
ZeigerAufBODY.replaceChild(ZeigerAufTRNeu, ZeigerAufTRAlt);
```

4.3.1.3.3.2. **HTML-DOM beim Internet Explorer**

4.3.1.3.3.2.1. **thematisierte Zuordnung von Eigenschaften und Methoden des HTML-DOM zu ausgewählten Objekten und zu Anwendungsbereichen im Internet Explorer (Übersicht)**

Attribut Objekt

Eigenschaft	.specified	prüfen ob ein Attribut im Element einen Wert hat, egal ob Element mit oder ohne HTML-Tag-Anweisung erzeugt wurde
Eigenschaft	.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
Eigenschaft	.value	Wert eines Attributes eines Objektes
Methode	.clearAttributes()	alle HTML-Attribute eines Objektes entfernen, außer ID, STYLE und per Script definierte Attribute
Methode	.createAttribute()	ein Attribut im Dokument erzeugen und Referenz liefern Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugten Attribute !
Methode	.getAttribute()	Wert eines Attributes, das durch HTML-Anweisungen erzeugt wurde, liefern
Methode	.getAttributeNode()	Referenz auf Eigenschaft des Attribute-Objektes liefern, also Zeiger auf attribute.name property. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie
Methode	.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und im Ziel mit den vorhandenen Attributen mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID und NAME auch ab NS 6.x
Methode	.removeAttribute()	entfernen von Attribut, das durch HTML-Anweisungen erzeugt wurde
Methode	.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
Methode	.setAttribute()	Wert von vorhandenem Attribut neu definieren wenn Attribut nicht vorhanden ist, so es erzeugen und mit Wert belegen
Methode	.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
Comment Objekt		
Methode	.createComment()	Comment-Objekt (Kommentar-Objekt) im Dokument erzeugen und Referenz liefern
Document Objekt		
Eigenschaft	.documentElement	Referenz auf Wurzelknoten (root node) des Dokumentes liefern
Eigenschaft	.ownerDocument	Referenz auf das document-Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
Eigenschaft	.XSLDocument	Referenz auf den obersten Knoten des XSL-Dokumentes (Style-Sheet-Dokument)
Methode	.createAttribute()	ein Attribut im Dokument erzeugen und Referenz liefern Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugten Attribute !
Methode	.createComment()	Comment-Objekt (Kommentar-Objekt) im Dokument erzeugen und Referenz liefern
Methode	.createElement()	HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern Achtung: Erzeugtes Objekt muss in DOM noch per Methode .insertBefore() bzw. .appendChild() eingereiht werden. Hinweis: Attribute mit der Methode .createAttribute() erzeugen auch ab NS 6.x



Methode	.createStyleSheet()	Style-Sheet-Objekt im Dokument erzeugen und Referenz liefern
Methode	.createTextNode()	Textelement im Dokument erzeugen und Referenz liefern nur Plain-Text, also keine HTML-Tags auch ab NS 6.x
Methode	.getElementById()	Referenz auf das im Dokument zuerst gefundene Objekt laut ID liefern wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenzieren auch ab NS 6.x
Methode	.getElementsByName()	Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME liefern auch ab NS 6.x
DOM als symbolisches Objekt		
Methode	.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
Element (Objekt/Knoten) im DOM		
Eigenschaft	.documentElement	Referenz auf Wurzelknoten (root node) des Dokumentes liefern
Eigenschaft	.nodeType	Knotentyp laut attributes Collection
Eigenschaft	.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
Eigenschaft	.ownerDocument	Referenz auf das document-Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
Eigenschaft	.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
Eigenschaft	.scopeName	Namensraum laut XMLNS-Attribut
Eigenschaft	.tagUrn	Uniform Ressource Name (URN) laut Namensraum laut XMLNS-Attribut
Eigenschaft	.uniqueID	durch den Browser automatisch generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
Eigenschaft	.XMLDocument	Referenz auf XML-Dokument (XML-DOM)
Methode	.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen ab IE 5.x bis unter IE 5.5
Methode	.cloneNode()	Objekt klonen und Referenz des Klone liefern
Methode	.getElementById()	Referenz auf das im Dokument zuerst gefundene Objekt laut ID liefern wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenziert auch ab NS 6.x
Methode	.getElementsByName()	Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME liefern auch ab NS 6.x
Methode	.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. auch ab NS 6.x
Methode	. implementation .hasFeature()	Objektzugehörigkeit zum DOM (Document Object Model-Standard) (HTML-DOM bzw. XML-DOM)
Methode	.insertAdjacentElement()	Element in eine Element einfügen und Referenz liefern wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben nur nach dem kompletten Laden des Dokumentes



Methode	.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
Methode	.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
Methode	.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
Methode	.swapNode()	Positionen von 2 Knoten in der Dokumenthierarchie tauschen nur sichtbar wenn Endetag geparkt

Eltern in der Vererbung

Eigenschaft	.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
Eigenschaft	.documentElement	Referenz auf Wurzelknoten (root node) des Dokumentes liefern
Eigenschaft	.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
Eigenschaft	.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM-Hierarchie
Eigenschaft	.parentTextEdit	Textbereich des Elternobjektes referenzieren
Methode	.applyElement()	Elementeigenschaft Kind oder Eltern festlegen und Referenz liefern Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht
Methode	.hasChildNodes()	prüfen auf Existenz von Kinder(n) von HTML-Elemente oder Textknoten

HTML-Element

Eigenschaft	.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
Eigenschaft	.documentElement	Referenz auf Wurzelknoten (root node) des Dokumentes liefern
Eigenschaft	.tagName	Tag-Bezeichner des Objektes
Methode	.clearAttributes()	alle HTML-Attribute eines Objektes entfernen, außer ID, STYLE und per Script definierte Attribute
Methode	.createElement()	HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern Achtung: Erzeugtes Objekt muss in DOM noch per Methode .insertBefore() bzw. .appendChild() eingereiht werden. Hinweis: Attribute mit der Methode .createAttribute() erzeugen auch ab NS 6.x
Methode	.expression()	Wert einer Style-Eigenschaft per STYLE-Attribut-Wert in HTML als Ausdruck definieren für Berechnung per Methode .getExpression() Ausdruck nur als Script kodierbar
Methode	.getAttribute()	Wert eines Attributes, das durch HTML-Anweisungen erzeugt wurde, liefern
Methode	.hasChildNodes()	prüfen auf Existenz von Kinder(n) von HTML-Elemente oder Textknoten
Methode	.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen nur nach dem kompletten Laden des Dokumentes HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so Einfügen nicht ausgeführt eingefügter Code wird sofort geparkt und ausgeführt bei Script-Code: <SCRIPT DEFER> muss kodiert werden
Methode	.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und im Ziel mit den vorhandenen Attributen mischen Attribute sind: HTML Events Styles



ab IE 5.01 auch ID und NAME

auch ab NS 6.x

Methode	.removeAttribute()	entfernen von Attribut, das durch HTML-Anweisungen erzeugt wurde
HTML und Script in einem Element		
Methode	.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen nur nach dem kompletten Laden des Dokumentes HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so Einfügen nicht ausgeführt eingefügter Code wird sofort geparkt und ausgeführt bei Script-Code: <SCRIPT DEFER> muss kodiert werden
Kind in der Vererbung		
Eigenschaft	.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
Eigenschaft	.documentElement	Referenz auf Wurzelknoten (root node) des Dokumentes liefern
Eigenschaft	.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
Eigenschaft	.lastChild	Zeiger auf das LETZTE Kind laut childNodes-Collection eines Objektes
Eigenschaft	.nextSibling()	Zeiger auf das NACHFOLGENDE Kind laut childNodes-Collection eines Objektes
Eigenschaft	.nodeName	String als Name des Kindes liefern, also TAG-Bezeichner, Attribut-Name, #text für Anker
Eigenschaft	.previousSibling	Zeiger auf das VORHERGEHENDE Kind
Methode	.appendChild()	Knoten als Kind an das DOM anhängen und Zeiger liefern Zeiger wird am Ende der Collection childNodes angehängen
Methode	.applyElement()	Elementeigenschaft Kind oder Eltern festlegen und Referenz liefern Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht
Methode	.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das Element Eltern hat und somit ein Kind ist
Methode	.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
Methode	.removeChild()	Kind entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
Methode	.replaceChild()	Kind ersetzen durch Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
Style Objekt		
Methode	.createStyleSheet()	Style-Sheet-Objekt im Dokument erzeugen und Referenz liefern
Methode	.expression()	Wert einer Style-Eigenschaft per STYLE-Attribut-Wert in HTML als Ausdruck definieren für Berechnung per Methode .getExpression() Ausdruck nur als Script kodierbar
Methode	.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
Methode	.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und im Ziel mit den vorhandenen Attributen mischen



		Attribute sind: HTML Events Styles ab IE 5.01 auch ID und NAME auch ab NS 6.x
Methode	.removeExpression()	Ausdruck für die Berechnung des Wert einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft.....</code> entfernen
Methode	.setExpression()	Wert einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft.....</code> anhand eines Ausdruck definieren für Berechnung per Methode <code>getExpression()</code> Ausdruck nur als Script kodierbar
Text Objekt		
Eigenschaft	.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
Methode	.createTextNode()	Textelement im Dokument erzeugen und Referenz liefern nur Plain-Text, also keine HTML-Tags auch ab NS 6.x
Methode	.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
Methode	.hasChildNodes()	prüfen auf Existenz von Kinder(n) von HTML-Elemente oder Textknoten
Methode	.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen nur nach dem kompletten Laden des Dokumentes
Methode	.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern

4.3.1.3.3.2.2. **Eigenschaften zur Verwaltung des HTML-DOM im Internet Explorer**

Nachfolgende Eigenschaften sind nicht in allen Objekten implementiert (siehe einzelne Objekte).

.canHaveChildren prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
Beispiel:

```

<SCRIPT>
function KindHinzufuegen()
{
    var ZeigerAufSPANObjekt = document.createElement("SPAN");

    var ZeigerAufTextObjekt = document.createTextNode("Test");

    ZeigerAufSPANObjekt.appendChild(ZeigerAufTextObjekt);

    for(var Index=0; Index <document.all.length; Index ++){
        if(document.all[Index].canHaveChildren==true)
        {
            document.all[Index].appendChild(ZeigerAufSPANObjekt);
            break;
        }
    }
}
</SCRIPT>
<INPUT TYPE=button VALUE="Kind hinzufügen " onclick="KindHinzufuegen()">
<DIV>
    Test<BR>
</DIV>

```

.canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function Antwort(BooleanWert)
{ BooleanWert ? alert("Ja") : alert("Nein"); }
</SCRIPT>

```



```

</HEAD>
<BODY>
  <P>
    INPUT:
    <INPUT type="text" ID="ID_Input" VALUE="Test" >
  </P>

  <P>
    SPAN:
    <SPAN ID="ID_Span">Test</SPAN>
  </P>

  <BUTTON onclick="Antwort(ID_Input.canHaveHTML);">
    Kann INPUT-Element HTML besitzen ?
  </BUTTON>

  &nbsp;

  <BUTTON onclick="Antwort(ID_Span.canHaveHTML);">
    Kann SPAN-Element HTML besitzen ?
  </BUTTON>
</BODY>
</HTML>

```

`.documentElement` Referenz auf Wurzelknoten (root node) des Dokumentes liefern

Beispiel:

```

<SCRIPT>
  function fnGetHTML()
  {ID_Textarea.value= document.documentElement.innerHTML;}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS = 50 ROWS = 10>
</TEXTAREA>

```

`.firstChild` Zeiger auf das ERSTE Kind laut `childNodes`-Collection eines Objektes

Beispiel:

```

<SCRIPT>
  var ZeigerAufErstesKind = Liste.firstChild; // liefert Referenz auf Listenelement 1
</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1
    <LI> Listenelement 2
    <LI> Listenelement 3
  </UL>
</BODY>

```

`.lastChild` Zeiger auf das LETZTE Kind laut `childNodes` Collection eines Objektes

Beispiel:

```

<SCRIPT>
  var ZeigerAufLetztesKind = Liste.lastChild; // liefert Referenz auf Listenelement 3
</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1
    <LI> Listenelement 2
    <LI> Listenelement 3
  </UL>
</BODY>

```

`.nextSibling` Zeiger auf das NACHFOLGENDE Kind laut `childNodes` Collection eines Objektes

Beispiel:

```

<SCRIPT>
  var ErstesKind_Index = 0; // Index ab 0
  var ZeigerAufNachfolgenKind = Liste.childNodes(ErstesKind_Index).nextSibling;
  // liefert Referenz auf Listenelement 2
</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1

```



```

        <LI> Listenelement 2
        <LI> Listelement 3
    </UL>
</BODY>

```

.nodeName String als Name des Kindes (Knoten, Node, Element)
 also TAG-Bezeichner, Attribut-Name; #text für Anker

Beispiel:

```

<SCRIPT>
    var ErstesKind_Index = 0;        // Index ab 0
    var ErstesKind_Name = Liste.childNodes(ErstesKind_Index).nodeName;
    alert(ErstesKind_Name);          // liefert den Tagnamen 'LI' von Listenelement 1
                                     // Hinweis: Listenelement 1 ist der Wert des Kindes
</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listelement 2
        <LI> Listelement 3
    </UL>
</BODY>

```

.nodeType Knotentyp laut attributes Collection

1	für Element-Knoten
2	für Attribut-Knoten
3	für Textknoten
4	für CDATA-Abschnitt
5	für Entity-Referenz
6	für Entity-Knoten
7	für Processing Instruction
8	für Kommentar-Knoten
9	für das Dokument
10	für den Dokumenttyp
11	für ein Dokument-Fragment
12	für Notation
null	wenn Knoten nicht vorhanden (null-Zeiger)

Beispiel 1:

```
var KnotenTypVonBODY = document.body.nodeType;
```

Beispiel 2:

```

var ZeigerAuf_B_Tag = document.createElement("B");        // B-Tag erzeugen
document.body.insertBefore(ZeigerAuf_B_Tag);
                                     // B-Tag in den BODY-Teil des Dokument einfügen,
                                     // also in die Baumstruktur
var KnotenTypDes_B_Tag = ZeigerAuf_B_Tag.nodeType;

```

.nodeValue Knotenwert (Wert des Kindes, Node, Elementes)
 nur für Text- und Attribut-Elemente
 nicht für Element-Knoten (Knotentyp 1)

Beispiel:

```

<SCRIPT>
    function KnotenWertAendern( ZeigerAufListe,
                                IndexVonListenelement, // immer ab 0
                                Zeichenkette             // Listenelement muss Text sein
                                )
    {
        var ReturnWert=false;        // Annahme: Änderung schlägt fehl

        // prüfen auf UL-Tag
        if (ZeigerAufListe.nodeName == "UL")
        {
            // Anzahl der Listenelemente holen
            var AnzahlListenelemente= ZeigerAufListe.childNodes.length;
                                                    // immer ab 1

            // und Anzahl und Index prüfen
            if ( (AnzahlListenelemente > 0)        // immer ab 1
                && (IndexVonListenelement >= 0)    // immer ab 0
                && (IndexVonListenelement < AnzahlListenelemente)
                                                    // Index ist zulässig zur Anzahl
            )
            {

```



```

// Zeiger auf das Listenelement laut Index holen
var ZeigerAufListenElement =
    ZeigerAufListe.childNodes(IndexVonListenElement);

// existiert das Listenelement ?
if (ZeigerAufListenElement)
{
    // ZeigerAufListenElement ist nicht null

    // Listenelement ist Textelement ?
    if (ZeigerAufListenElement.nodeType == 3)
    {
        ZeigerAufListenElement.nodeValue =
            Zeichenkette;

        ReturnWert =true;
    }
}

return ReturnWert;
}
</SCRIPT>
<UL ID="Liste" onclick=" KnotenWertAendern(this, 0, 'Listenelement Neu')">
    <LI>Listenelement alt
</UL>

```

.ownerDocument Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde

Beispiel:

```

<SCRIPT>
    var ElternDokument = SpanTag.ownerDocument;
</SCRIPT>
<BODY>
    <SPAN ID="SpanTag">Hallo !</SPAN>
</BODY>

```

.parentElement Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
 var Zeiger = document.body.parentElement ;

.parentNode Referenz auf Elternknoten innerhalb der DOM-Hierarchie

Beispiel 1:

```

<SCRIPT>
    var ElternZeiger = SpanTag.parentNode;
</SCRIPT>
<BODY>
    <SPAN ID="SpanTag">Hallo !</SPAN>
</BODY>

```

Beispiel 2:

document.body. parentNode wird null liefern, da BODY das oberste Objekt

Beispiel 3:

```

var ZeigerAuf_B_Tag = document.createElement("B");           // erzeugen
document.body.insertBefore(ZeigerAuf_B_Tag);                 // einfügen
var ElternZeiger = ZeigerAuf_B_Tag.parentNode;                 // Zeiger auf BODY

```

.parentTextEdit Textbereich des Elternobjektes referenzieren

Beispiel:

```

<SCRIPT>
    function Selektion()
    {
        var ZeigerAufSelektionsQuelle = window.event.srcElement ;

        if (!ZeigerAufSelektionsQuelle.isTextEdit)
        { ZeigerAufSelektionsQuelle = ZeigerAufSelektionsQuelle.parentTextEdit;}

        if (ZeigerAufSelektionsQuelle != null)
        {

```



```

        var ZeigerAufTextBereich =
            ZeigerAufSelektionsQuelle.createTextRange();

        ZeigerAufTextBereich.moveToElementText(window.event.srcElement);
        ZeigerAufTextBereich.collapse();
        ZeigerAufTextBereich.expand("SelektionsText");
        ZeigerAufTextBereich.select();
    }
}
</SCRIPT>

```

.previousSibling Referenz auf das Vorgängerkind

Beispiel:

```

<SCRIPT>
    var AktuellesKind_Index = 1;    // Index ab 0
    var VorgaengerKind_Zeiger = Liste.childNodes(AktuellesKind_Index).previousSibling;
                                // Listenelement 1 wird referenziert
</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listenelement 2
        <LI> Listenelement 3
    </UL>
</BODY>

```

.readyState aktueller Status des Objektes beim Füllen des Objektes mit Daten

"uninitialized"	Objekt ist nicht initialisiert
"loading"	Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung
"loaded"	Objekt hat alle Daten komplett geladen und ist komplett initialisiert
"interactive"	Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten
"complete"	Object hat alle Daten geladen und ist mit diesen komplett initialisiert

Beispiel:

```
alert(document.body.readyState);
```

.scopeName Namensraum laut XMLNS-Attribut

Beispiel:

```

<HTML XMLNS:InetSDK='http://www.test.de'>
<STYLE>
    @media all {InetSDK:HalloDu { behavior:url (simple.htc) }}
</STYLE>
<SCRIPT>
    function window.onload()    // überschreibt Standard window.onload-Routien !!!!
    {
        // Statuszeile als Ausgabebereich nutzen
        window.status = 'scopeName = ' + Hallo.scopeName + ';'
                        + ' tagUrn = ' + Hallo.tagUrn;
    }
</SCRIPT>
<BODY>
    <InetSDK:HelloWorld ID=Hallo></InetSDK:HalloDu>
</BODY>
</HTML>

```

.specified prüfen ob Objekt Attribute hat
 true, so Attribute ist vorhanden
 false, so keine Attribute vorhanden

Beispiel:

```

<SCRIPT>
    function Anzeigen()
    {
        var FeldAllerAttribute = ID_List.attributes;
        alert(FeldAllerAttribute(0).nodeName);    // Knotenname

        for( var i=0; i< FeldAllerAttribute.length; i++)
        {
            // neues LI-Element als Knoten der Liste anhängen
            var NeuesListenElement = document.createElement("LI");
            ID_List.appendChild(NeuesListenElement);
        }
    }

```



```
// Wert des neuen LI-Elementes ist Text, also Textknoten
var WertNeuesListenElement = document.createTextNode(
    i
    + " "
    + FeldAllerAttribute(i).nodeName
    + " = "
    + FeldAllerAttribute(i).nodeValue
);
NeuesListenElement.appendChild(WertNeuesListenElement);

// auf specified prüfen
if(FeldAllerAttribute(i).nodeValue != null )
{
    alert(    FeldAllerAttribute(i).nodeName
            + " specified: "
            + FeldAllerAttribute(i).specified // boolean
    );
}
}
```

```
</SCRIPT>
<UL ID="ID_List" onclick = " Anzeigen()">
    <LI>Klick mich
</UL>
```

.tagName Tag-Bezeichner des Objektes

Beispiel:

```
<SCRIPT>
var ID_Wert = window.prompt("Bitte ID eines Tags eingeben:");
if (ID_Wert != null)
{alert(document.all[ID_Wert].tagName) }
</SCRIPT>
```

.tagUrn Uniform Ressource Name (URN) laut Namensraum laut XMLNS-Attribut

Beispiel:

```
<HTML XMLNS:InetSDK='http://www.test.de'>
<STYLE>
    @media all {InetSDK\;HalloDu { behavior:url (simple.htc) }}
</STYLE>
<SCRIPT>
function window.onload() // überschreibt Standard window.onload-Routien !!!!
{
    // Statuszeile als Ausgabebereich nutzen
    window.status = 'scopeName = ' + Hallo.scopeName + ' ;'
                  + ' tagUrn = ' + Hallo.tagUrn;
}
</SCRIPT>
<BODY>
    <InetSDK:HelloWorld ID="Hallo"></InetSDK:HalloDu>
</BODY>
</HTML>
```

.uniqueID durch den Browser automatisch generiertes ID des Objektes
Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

Beispiel:

```
<PUBLIC:ATTACH EVENT="onload" FOR="window" ONEVENT="init()"/>
<SCRIPT LANGUAGE="JScript">
function init()
{
    var newTextAreaID = element.document.uniqueID;
    element.document.body.insertAdjacentHTML ( "beforeEnd",
        "<P><TEXTAREA STYLE='height: 200 ;'"
        + "width: 350' ID=" + newTextAreaID
        + "></TEXTAREA></P>"
    );
}
</SCRIPT>
```

.value Wert eines Objekt-Attributes



Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar

Beispiel für Input-Objekt und seine Varianten:

für Input-Elemente (input Objekt) gelten folgende Standardwerte:

INPUT type=checkbox	on
INPUT type=reset	Reset
INPUT type=submit	Submit Query

alle anderen Input-Elemente haben keinen Standardwert

für Input-Elemente (input Objekt) sind folgenden Werte sendbar:

INPUT type=checkbox	selektierter Wert
INPUT type=file	Dateiname laut Eingabe
INPUT type=hidden	selektierter Wert einer Box-Control

(selektierte

Option)

INPUT type=password	Eingabewert
INPUT type=radio	selektierter Wert
INPUT type=reset	das Label des Elementes (falls Label existent

ist)

INPUT type=submit	das Label des Elementes (falls Label existent
-------------------	--

ist)

INPUT type=text	Eingabewerte
-----------------	--------------

Werte sind les- und schreibbar

nur lesen bei INPUT type=file

Beispiel für option objekt eines select objektes:

```
<SCRIPT>
function Anzeigen()
{
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
}
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
<OPTION VALUE="1">Auswahl 1 </OPTION>
<OPTION VALUE="2">Auswahl 2 </OPTION>
<OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>
```

Beispiel für Zeichenkette auf Wertebereich der Zeichen prüfen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function pruefe_zeichenkette(zeichenkette, wertebereich)
{
    // wertebereich ist Zeichenkette z.B. "0123456789 -+/,.)"

    var rueckgabewert = true;    // Annahme: Zeichenkette hat NUR Zeichen
                                // aus dem Wertebereich

    var zeichen_aus_zeichenkette;

    //      Zeichenkette zeichenweise analysieren: Jedes Zeichen mit Wertebereich vergleichen
    for (var i = 0; i < zeichenkette.length; i++)
    {
        zeichen_aus_zeichenkette = zeichenkette.charAt(i);

        if (wertebereich.indexOf(zeichen_aus_zeichenkette) == -1)
        {
            rueckgabewert = false;
            break;    // ungültiges Zeichen gefunden, also abbrechen
        }
    }

    return rueckgabewert;
}
```




```

function pruefe_eingabe(zu_pruefende_zeichenkette)
{
    if (pruefe_zeichenkette(zu_pruefender_zeichenkette, "0123456789 -+/,()") )
    {alert("Eingabe ist korrekt !");}
    else
    {alert("Eingabe ist nicht korrekt !"); }
}

//-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    Telefon:
    <INPUT  TYPE="text"
           NAME="Telefon"
           VALUE=""
    >
    <INPUT  TYPE="button"
           VALUE="Ueberpruefen"
           onclick="pruefe_eingabe(this.form.Telefon.value)">

</FORM>
</BODY>
</HTML>

```

.XMLDocument Referenz auf XML-Dokument (XML-DOM)

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    var ZeigerAufXMLDocument = ID_Div.XMLDocument;
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div">
    </DIV>
</BODY>
</HTML>

```

.XSLDocument Referenz auf den obersten Knoten des XSL-Dokumentes (Style-Sheet-Dokument)
 var Zeiger = document.XSLDocument;

4.3.1.3.3.2.3. Methoden zur Verwaltung des HTML-DOM im Internet Explorer

Nachfolgende Methoden sind nicht in allen Objekten implementiert (siehe einzelne Objekte).

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen
 Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
 ab IE 5.x bis unter IE 5.5

Beispiel:

```

<SCRIPT>
    var FeldDerEigenschaftenID                      = new Array(); // für removeBehavior
    var FeldDerTagsLlimDokument                      = new Array ();
    var FeldDerTagsLlimDokument_Laenge = 0;

    function EigenschaftHinzufuegen()
    {
        FeldDerTagsLlimDokument                      = document.all.tags ("LI");
        FeldDerTagsLlimDokument_Laenge = FeldDerTagsLlimDokument.length;
        for ( var i=0; i < FeldDerTagsLlimDokument_Laenge; i++)
        {
            var EigenschaftenID    // immer neu anlegen wegen Zeigerprüfung
            = FeldDerTagsLlimDokument [i].addBehavior ("hilite.htc");

            if (iEigenschaftenID)
            {FeldDerEigenschaftenID[i] = EigenschaftenID;}
        }
    }

    function EigenschaftEntfernen()
    {
        for ( var i=0; i < FeldDerTagsLlimDokument_Laenge; i++)

```



```

    }
    {FeldDerEigenschaftenID[i].removeBehavior (FeldDerEigenschaftenID[i]); }
  }
</SCRIPT>
<A HREF="javascript: EigenschaftHinzufuegen()">Eigenschaft hinzufuegen</A>
<A HREF="javascript: EigenschaftEntfernen()">Eigenschaft entfernen</A>

```

.appendChild() Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
 DOM wird geändert
 Zeiger wird zugleich immer am Ende der Collection `childNodes` angehängen
 Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
 kann erst nach `createElement()` erfolgen
`.readyState` wird erst belegt mit `.appendChild()`

Beispiel 1:

```

var ZeigerAufDiv =document.createElement("DIV");
document.body.appendChild(ZeigerAufDiv);

```

Beispiel 2:

```

<SCRIPT>
function Anhaengen()
{
    var ZeigerAufNeuesLI = document.createElement("LI");
    Liste.appendChild(ZeigerAufNeuesLI);           // dem BODY anhängen
                                                    // also sichtbar werdend
    ZeigerAufNeuesLI.innerHTML="Listenelement 5";
}
</SCRIPT>
<BODY>
    <UL ID = "Liste" >
        <LI>Listenelement 1
        <LI>Listenelement 2
        <LI>Listenelement 3
        <LI>Listenelement 4
    </UL>
    <INPUT TYPE = "button" VALUE = "Anhaengen " onclick = " Anhaengen()">
</BODY>

```

.applyElement() Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
 DOM wird geändert
 Element kann selbst Kinder haben
 Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
 Achtung: Wenn Element per Methode `.createElement()` erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft `.innerHTML` gelöscht !

Beispiel:

```

<SCRIPT>
function Hinzufuegen()
{
    var Zeiger = document.createElement("LI");
    Liste.applyElement(Zeiger);
}
</SCRIPT>
<UL ID = Liste>
    <LI>Listenelement 1
    <LI>Listenelement 2
    <LI>Listenelement 3
    <LI>Listenelement 4
</UL>
<INPUT TYPE="button" VALUE=" Hinzufuegen" onclick=" Hinzufuegen()">

```

.clearAttributes() alle HTML-Attribute eines Objektes entfernen, außer ID, STYLE und per Script definierte Attribute
 Script-erzeugte Attribute nicht entfernen
 DOM wird geändert

Beispiel:

```

<SCRIPT>
function Loeschen()
{
    ID_Span.children[0].clearAttributes();
}
</SCRIPT>

```



```

<SPAN ID="ID_Span">
  <DIV ID="ID_Div"
    ATTRIBUTE1="true"
    ATTRIBUTE2="true"
    onclick="alert('click');"
    onmouseover="this.style.color='#0000FF';"
    onmouseout="this.style.color='#000000';"
  >
    Test eines<B>Div</B>Elementes.
  </DIV>
</SPAN>
<INPUT TYPE="button" VALUE=" Loeschen" onclick="Loeschen()">

```

.cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
 DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)

Beispiel:

```

<SCRIPT>
function Klonen()
{
    var ZeigerAufKlone = Liste.cloneNode(true); // DOM wird nicht geändert
    document.body.insertBefore(ZeigerAufKlone); // DOM wird geändert
}
</SCRIPT>
<UL ID = "Liste" >
  <LI>Listenelement 1
  <LI>Listenelement 2
  <LI>Listenelement 3
  <LI>Listenelement 4
</UL>
<INPUT TYPE="button" VALUE=" Klonen" onclick=" Klonen()">

```

.contains() prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
 DOM nicht geändert
 true Element liegt innerhalb eines Elementes
 false Element liegt nicht innerhalb eines Elementes

Beispiel:

```

<SCRIPT>
function TesteMaus(Zeiger)
{
    if( ! Zeiger.contains(event.fromElement) )
    {alert("Maus über Button erkannt");}
}
</SCRIPT>
<BUTTON onmouseover=" TesteMaus(this)">Ueberfahre mich mit der Maus</BUTTON>

```

.createAttribute() ein Attribut im Dokument erzeugen und Referenz auf das erzeugte Attribut liefern
 Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
 DOM nicht geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function Hinzufuegen()
{
    // Attribut mit Wert erzeugen
    var ZeigerAufAttribut = document.createAttribute("title");
    ZeigerAufAttribut.value = "Tooltip-Text ";

    // Attribut als Feldelement anhängen
    var ZeigerAufFeld = ID_Div.attributes;
    ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
}
</SCRIPT>
</HEAD>
<BODY onload=" Hinzufuegen();">
  <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```



`.createComment()` Comment-Objekt (Kommentar-Objekt) im Dokument erzeugen und Referenz liefern
verwendbar anstelle von direkt im HTML- bzw. Script-Quellcode kodiertem Kommentar
DOM wird geändert, da das Dokument erweitert wird

Beispiel:

```
var Kommentar = document.createComment("Das ist ein Kommentar");
```

`.createElement()` HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern
Achtung: Erzeugtes Objekt muss in DOM noch per Methode `.insertBefore()` bzw. `.appendChild()` eingereiht werden.
Hinweis: Attribute mit der Methode `.createAttribute()` erzeugen
DOM wird geändert

nach `createElement()` muss irgendwann z.B. `appendChild()` folgen

`.readyState` wird erst belegt mit `.appendChild()`

Tags sind auch in der Form '`< ...>`' mit Attributen möglich

Ende-Tag kann muss aber nicht kodiert werden (falls HTML-Syntax einen Endetag vorschreibt)

Bsp: '`<DIV STYLE="...">`' oder '`<DIV STYLE="..."></DIV>`'

niemals Elemente zwischen den Tags angeben sondern diese NACH der Erzeugung
per `.innerText` bzw. `.innerHTML` erzeugen

Achtung: beide Eigenschaften schalten aktives BGSOUND auf stumm !

also z.B. nicht '`<DIV STYLE="...">TestText</DIV>`'

'`TestText</DIV>`'

es werden z.B. HTML-Code-Fehler ignoriert, aber nicht alle

wenn ignoriert, so Objekt erzeugt aber eben nicht mit vollständigem HTML-Code

Beispiel 1:

```
<SCRIPT>
function Erzeugen()
{
    ID_Span.innerHTML="";
    var FeldDerZeigerAufOption = ID_Select.options[ID_Select.selectedIndex];

    if(FeldDerZeigerAufOption.text.length>0)
    {
        var ZeigerAufElement=
            document.createElement(FeldDerZeigerAufOption.text);
        eval(
            " ZeigerAufElement."
            + FeldDerZeigerAufOption.value
            + "="
            + ID_Input.value
            + ""
        );

        if(FeldDerZeigerAufOption.text=="A")
        { ZeigerAufElement.href="javascript:alert('A link.');" }

        ID_Span.appendChild(ZeigerAufElement);
    }
}
</SCRIPT>
<SELECT ID="ID_Select" onchange="Erzeugen()">
    <OPTION VALUE="innerText">A
    <OPTION VALUE="value">&lt;INPUT TYPE="button"&gt;
</SELECT>
<INPUT TYPE="text" ID="ID_Input" VALUE="Beispiel Text">
<SPAN ID="ID_Span" ></SPAN>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
function Erzeuge()
{
    var Zeiger = document.createElement(
        "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Eins'>"
    );
    document.body.insertBefore(Zeiger);

    Zeiger = document.createElement(
        "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Zwei'>"
    );
    document.body.insertBefore(Zeiger);
}
```



```

    }
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE="BUTTON"
           ONCLICK=" Erzeuge()"
           VALUE="Zwei Radio Buttons erzeugen">

    <BR>
    <INPUT TYPE="BUTTON"
           ONCLICK="alert(document.body.outerHTML)"
           VALUE="Click um HTML zu sehen">

    <BODY>
</HTML>

```

.createStyleSheet() Style-Sheet-Objekt im Dokument erzeugen und Referenz liefern
DOM wird geändert

Beispiel:
document.createStyleSheet('styles.css');

.createTextNode() Plain-Textelement im Dokument erzeugen und Referenz liefern
Plaintext kann keine HTML-Tags enthalten
DOM wird geändert, da Dokument erweitert wird

Beispiel:

```

<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>

```

.expression() Wert einer Style-Eigenschaft per STYLE-Attribut-Wert in HTML als Ausdruck definieren für
spätere Berechnung per Methode.getExpression()
Ausdruck nur als Script kodierbar
DOM wird geändert
siehe auch Methode .setExpression()

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE = "JScript">
function width_init()
{
    ID_Span.style.setExpression( "width",
                                " trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                ",
                                "jscript"
                                );
}

function Berechne()
{alert(ID_Span.style.getExpression("width");}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
        STYLE="background-color:lightgreen;
               width:expression( trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                )
               "
        >
    </SPAN>
    <BUTTON onclick= Berechne();>
</BODY>
</HTML>

```



Beispiel 2:

```

ID_Span.style.setExpression("height","document.style.fontSize + 13");
ID_Span.style.setExpression("width","document.body.style.fontSize");
<SPAN ID="ID_Span"
      STYLE="background-color:lightgreen;
            width:expression( trueBlueSpan.style.pixelWidth
                              + oldYellowSpan.style.pixelWidth
                              )
            "
>
</SPAN>

```

`.getAdjacentText()` Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
Text kann HTML-Tags enthalten, muss aber nicht
DOM nicht geändert

Beispiel:

```

<SCRIPT>
function Anzeigen()
{
    var Woher_Kette = ID_Selektion.options[ID_Selektion.selectedIndex].text;
    alert(ID_P.getAdjacentText(Woher_Kette));
}
</SCRIPT>
Text vor dem Beginn des P-Tag (woher ist beforeBegin)
<P ID="ID_P">
    Text nach dem Beginn des P-Tag (woher ist afterBegin)
    <B>Irgend ein Text</B>
    Text vor dem Ende des P-Tag (woher ist beforeEnd)
</P>
Text nach dem Ende des P-Tag (woher ist afterEnd)

<SELECT ID="ID_Selektion">
    <OPTION SELECTED>woher ist beforeBegin
    <OPTION>woher ist afterBegin
    <OPTION>woher ist beforeEnd
    <OPTION>woher ist afterEnd
</SELECT>
<INPUT TYPE="button" VALUE="Anzeigen" onclick=" Anzeigen()">

```

`.getAttribute()` Wert eines per **HTML** erzeugten Attributes liefern
DOM nicht geändert

Beispiel:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachenames definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;
    }

```



```

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

`.getAttributeNode()` Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf `attribute.name` Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht. Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt. Wert des Attributes wird somit über die Referenz laut DOM erreichbar. DOM nicht geändert.

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function ToolTipKnotenErmitteln()
    {
        return (ID_Div.getAttributeNode("TITLE"));
    }
</SCRIPT>
</HEAD>
<BODY onload="ToolTipKnotenErmitteln();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

`.getElementById()` Referenz auf das im Dokument ZUERST gefundene Objekt laut ID (analog zum ID-Attribut) liefern. Achtung: Objekte, die kein ID besitzen, werden nicht erfasst!

- Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode `getElementsByName()`
- Für Verwaltung per Tag-Name: siehe Methode `getElementsByTagName()`

wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenziert

Eine Referenzierung einer Collection der Objekte mit gemeinsamen ID ist leider nicht möglich. Deswegen der strenge Hinweis:

In der Regel werden ID vom Programmierer objektweise getrennt vergeben, es sei denn, man will bewusst eine Gruppe von Objekten (z.B. RadioBox) verwaltbar



machen und kennt diese Objekte. Die maschinelle Analyse eines fremden Dokumentes mit der Methode `getElementById()` ist nicht möglich.

DOM nicht geändert

Beispiel:

```
<SCRIPT>
function Referenziere()
{
    var Zeiger = document.getElementById("ID_Div");
}
</SCRIPT>
<DIV ID="ID_Div">Test</DIV>
<INPUT TYPE="button" VALUE="Referenziere" onclick="Referenziere()">
```

`.getElementsByName()` Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME (analog zum Attribut NAME) liefern

Achtung: Objekte, die kein NAME besitzen, werden nicht erfasst !

Für Verwaltung per ID (analog zum ID-Attribut):

siehe Methode `getElementById()`

Für Verwaltung per Tag-Name

siehe Methode `getElementsByTagName()`

DOM nicht geändert

Beispiel:

```
<SCRIPT>
function Referenziere()
{
    var FeldDerInput = document.getElementsByName("GemeinsamerName");
}
</SCRIPT>
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="button" NAME="Button" VALUE="Referenziere" onclick="Referenziere()">
```

`.getElementsByTagName()` Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.

Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden

(beachte dabei `document.all-Collection`)

Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !

Für Verwaltung per ID (analog zum ID-Attribut):

siehe Methode `getElementById()`

Für Verwaltung per NAME (analog zum NAME-Attribut):

siehe Methode `getElementsByName()`

DOM nicht geändert

Beispiel 1:

```
var DIV_KinderZeigerFeld = document.body.getElementsByTagName("DIV");
```

Hinweis: entspricht `var DIV_KinderZeigerFeld = document.body.all.tags("DIV");`

Beispiel 2:

```
<SCRIPT>
var Feld_Span = ID_DivEltern.getElementsByTagName("SPAN");
// alle SPAN-Kinder referenzieren
</SCRIPT>
<DIV ID="ID_DivEltern">
    <SPAN>
        Span-Kind von ID_DivEltern
    </SPAN>
    <DIV>
        Div-Kind vonDivEltern-Span
        <SPAN>
            Span-Kind vonDivEltern-Span-Div
        </SPAN>
    </DIV>
</DIV>
```

Beispiel 3:

```
<SCRIPT>
function Anzeige()
{
    var ZeigerAufOnClickEventQuelle=event.srcElement;
    var Feld =
        ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
```




```

        alert(      "Anzahl LI : "
                    + Feld.length
                    + "\nErster Eintrag: "
                    + Feld [0].childNodes[0].nodeValue
                    );
    }
</SCRIPT>
<UL onclick="Anzeige()">
    <LI>Menuepunkt 1
    <UL>
        <LI> Menuepunkt 1.1
        <OL>
            <LI> Menuepunkt 1 1.1
            <LI> Menuepunkt 1 1.2
        </OL>
        <LI> Menuepunkt 1.2
        <LI> Menuepunkt 1.3
    </UL>
    <LI> Menuepunkt 2
    <UL>
        <LI> Menuepunkt 2.1
        <LI> Menuepunkt 2.3
    </UL>
    <LI> Menuepunkt 3
</UL>

```

.getExpression() Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern
 Style-Eigenschaft ist per Methoden
 expression() oder setExpression()
 zu definieren
 DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout
 (nach dem eventuellen expliziten Dokument-Refresh)

In nachfolgenden Beispielen wurde aus Platzgründen die Zeichenkette von STYLE umgebrochen,
 was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

Beispiel 1:

```

<SPAN ID="ID_Span"
      STYLE= "background-color:lightgreen;
              width:expression( trueBlueSpan.style.pixelWidth
                               + oldYellowSpan.style.pixelWidth
                               )
            "
>
</SPAN>
<BR>
<BUTTON onclick=alert(ID_Span.style.getExpression("width"));>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE = "JScript">
    function width_init()
    {
        ID_Span.style.setExpression("width",
                                     "
                                     trueBlueSpan.style.pixelWidth
                                     + oldYellowSpan.style.pixelWidth
                                     ",
                                     "jscript"
                                     );
    }

    function Berechne()
    {alert(ID_Span.style.getExpression("width"));}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
        STYLE="background-color:lightgreen;

```



```

        width:expression( trueBlueSpan.style.pixelWidth
                          + oldYellowSpan.style.pixelWidth
                          )
    "
>
</SPAN>
<BUTTON onclick= Berechne();>
</BODY>
</HTML>

```

.hasChildNodes() prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
DOM nicht geändert

Beispiel:

```

<HTML>
<BODY onload="alert(ID_Div.hasChildNodes());">
  <DIV ID="ID_Div" TITLE="Tooltip-Text">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

.insertAdjacentElement() Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann
wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM
nur nach dem kompletten Laden des Dokumentes möglich
DOM wird geändert

Beispiel:

```

<SCRIPT>
function Hinzufuegen()
{
    var ZeigerAufLI = document.createElement("LI");
    ID_Liste.children(0).insertAdjacentElement("beforeBegin", ZeigerAufLI);
    ZeigerAufLI.innerText = "Listeneintrag 0";
}
</SCRIPT>
<BODY>
  <OL ID = "ID_Liste ">
    <LI>Listeneintrag 1</LI>
    <LI>Listeneintrag 2</LI>
    <LI>Listeneintrag 3</LI>
  </OL>
  <INPUT TYPE = "button" VALUE = "Hinzufuegen" onclick=" Hinzufuegen()">
</BODY>

```

.insertAdjacentHTML() HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann
nur nach dem kompletten Laden des Dokumentes möglich
HTML- und Script-Code müssen syntaktisch korrekt sein
wenn nicht, so wird das Einfügen **nicht** ausgeführt
eingefügter Code wird **nur** dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist
bei Script-Code: <SCRIPT DEFER> muss kodiert werden
DOM wird geändert

Beispiel:

```

var HTML_Code =      "<INPUT TYPE =button"
                    +      " VALUE='Bitte klicken'"
                    +      " onclick='Anzeige()'"
                    +      ">"
                    +      "<BR>";

var JavaScript_Code = "<SCRIPT DEFER>" // DEFER muss kodiert sein
                    + "function Anzeige(){alert('Anzeige aktiv') }"
                    + "</SCRIPT>";

ID_Div.insertAdjacentHTML("afterBegin", HTML_Code + JavaScript_Code);

<DIV ID="ID_Div">
  Test
</DIV>

```

.insertAdjacentText() Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann
nur nach dem kompletten Laden des Dokumentes
DOM wird geändert

Beispiel:

```
var PlainText = " Hinzugefuegter Text";
```



```
ID_Div.insertAdjacentText("afterBegin", PlainText);
```

```
<DIV ID="ID_Div">
  Test
</DIV>
```

.insertBefore() Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode `createElement()` erzeugt worden sein
Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten
Sichtbarkeit erst wenn Ende-Tag geparkt wurde
DOM wird geändert

Beispiel:

```
<SCRIPT>
function Einfuegen()
{
    var ZeigerAufNeuesLI = document.createElement("LI");
    ID_UL.insertBefore(ZeigerAufNeuesLI, ID_LI);
    ZeigerAufNeuesLI.innerText="2";
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN onclick= Einfuegen()>Klick</SPAN>
    <UL ID="ID_UL">
        <LI >1</LI>
        <LI ID="ID_LI">3</LI>
        <LI >4</LI>
    </UL>
</BODY>
```

.mergeAttributes() alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
Attribute sind: HTML
Events
Styles
ab IE 5.01 auch ID, NAME
Achtung: Diese Methode ist mir Vorsicht zu genießen !!
DOM wird geändert

Beispiel:

```
<SCRIPT>
function Mischen()
{ ID_SPAN.children[1].mergeAttributes(ID_SPAN.children[0]);}
</SCRIPT>
<SPAN ID=ID_SPAN>
    <DIV ID="ID_Div_Quelle"
        ATTRIBUTE1="true"
        ATTRIBUTE2="true"
        onclick="alert('click');"
        onmouseover="this.style.color='#0000FF';"
        onmouseout="this.style.color='#000000';"
    >
        Quell<B>Div</B>
    </DIV>
    <DIV ID="ID_Div_Ziel">
        Ziel-Div
    </DIV>
</SPAN>

<INPUT TYPE="button" VALUE=" Mischen" onclick="Mischen()">
```

.normalize() Normalisierung des DOM zur Erreichung einer konsistenten Struktur
Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen

Beispiel:

```
<HTML>
<BODY onload="ID_Div.normalize();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>
```

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode `.createAttribute()` erzeugte Attribute werden nicht erfasst



DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Entfernen()
    {
        ID_Div.removeAttribute("TITLE");
    }
</SCRIPT>
</HEAD>
<BODY onload="Entfernen();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

.removeAttributeNode() entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function ToolTipKnotenEntfernen()
    {
        var Knoten = ID_Div.getAttributeNode("TITLE");
        ID_Div.removeAttributeNode(Knoten);
    }
</SCRIPT>
</HEAD>
<BODY onload="ToolTipKnotenEntfernen();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

.removeBehavior() per Methode **.addBehavior()** einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
DOM wird geändert

Beispiel:

```

<SCRIPT>
var FeldDerEigenschaftenID      = new Array(); // für removeBehavior
var FeldDerTagsLlimDokument     = new Array ();
var FeldDerTagsLlimDokument_Laenge = 0;

function EigenschaftHinzufuegen()
{
    FeldDerTagsLlimDokument      = document.all.tags ("LI");
    FeldDerTagsLlimDokument_Laenge = FeldDerTagsLlimDokument.length;
    for (i=0; i < FeldDerTagsLlimDokument_Laenge; i++)
    {
        var EigenschaftenID // immer neu anlegen wegen Zeigerprüfung
        = FeldDerTagsLlimDokument [i].addBehavior ("hilitc.htc");

        if (iEigenschaftenID)
        {FeldDerEigenschaftenID[i] = EigenschaftenID;}
    }
}

function EigenschaftEntfernen()
{
    for (i=0; i < FeldDerTagsLlimDokument_Laenge; i++)
    {FeldDerEigenschaftenID[i].removeBehavior(FeldDerEigenschaftenID[i]); }
}
</SCRIPT>
<A HREF="javascript: EigenschaftHinzufuegen()">Eigenschaft hinzufuegen</A>
<A HREF="javascript: EigenschaftEntfernen()">Eigenschaft entfernen</A>

```

.removeChild() Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern
Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
DOM wird geändert

Beispiel:



```

<HEAD>
<SCRIPT>
    function Entfernen()
    {
        // versuche den Text zu entfernen
        try
        {
            var KindZeigerAufTextImDiv = ID_Div.children(0);
            ID_Div.removeChild(KindZeigerAufTextImDiv);
            // Achtung: Der Text ist noch sichtbar !!!!
        }
        // oder fange das Ereignis des bereits entfernten Textes ein
        // und behandle das Ereignis
        catch(x)
        {
            alert(
                "Text wurde entfernt !\n"
                + "Das Dokument muss neu geladen werden !
            );
            document.location.reload();
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick=" Entfernen()">
        Klick, um diesen Text zu entfernen !
    </DIV>
</BODY>

```

.removeExpression() Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form `objekt.style.eigenschaft` dient.
 Ausdruck muss mit der Methode `.setExpression()` gesetzt worden sein
 DOM wird nicht geändert

Beispiel: aus Platzgründen die Zeichenkette von STYLE umgebrochen,
 was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

```

ID_Span.style.setExpression("width","document.body.style.fontSize");
<SPAN ID="ID_Span"
    STYLE="background-color:lightgreen;
        width:expression(
            trueBlueSpan.style.pixelWidth
            + oldYellowSpan.style.pixelWidth
        )
">
</SPAN>
ID_Span.style.removeExpression("width");

```

.removeNode() Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern
 Sichtbarkeit erst wenn Ende-Tag geparkt wurde
 DOM wird geändert

Beispiel:

```

<SCRIPT>
    function Entfernen()
    {Tabelle.removeNode(true);}
</SCRIPT>
<TABLE ID = "Tabelle" >
<TR>
    <TD>Zelle 1</TD>
    <TD>Zelle 2</TD>
</TR>
</TABLE>
<INPUT TYPE = button VALUE = " Entfernen" onclick = " Entfernen()">

```

.replaceAdjacentText() Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
 DOM wird nicht geändert

Beispiel:

```

var PlainText = "Neuer Text";
ID_Div.replaceAdjacentText("afterBegin", PlainText);

```



```
<DIV ID="ID_Div">
  Test
</DIV>
```

`.replaceChild()` Kind-Objekt ersetzen durch ein Objekt
 ersetzende Objekt muss per Methode `.createElement()` erzeugt worden sein
 Sichtbarkeit erst wenn Ende-Tag geparkt wurde
 DOM wird geändert

Beispiel:

```
<HEAD>
<SCRIPT>
  function Ersetze()
  {
    var KindZeigerAufDivText = ID_Div.children(0);
    var RetteInnerHTML = KindZeigerAufDivText.innerHTML;

    // prüfen auf Tag im Div-Text
    if (KindZeigerAufDivText.tagName=="B")
    {
      // Bold-Tag <B>gefunden, also I-Tag erzeugen
      var ZeigerAufNeuenSchriftStilTag =document.createElement("I");

      // komplettes ersetzen von Div-Text,
      ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
      ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
    else
    {
      // keinen Bold-Tag <B>gefunden
      var ZeigerAufNeuenSchriftStilTag =document.createElement("B");

      // komplettes ersetzen von Div-Text,
      ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
      ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
  }
</SCRIPT>
</HEAD>
<BODY>
  <DIV ID="ID_Div" onclick=" Ersetze()">
    Klicke für den Wechseln des <B>Schriftstils<B>
  </DIV>
</BODY>
```

`.replaceNode()` Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
 sichtbar erst mit parsen des Endetags
 DOM wird geändert

Beispiel:

```
<SCRIPT>
  function Ersetze()
  {
    var RetteInnerHTML = Liste.innerHTML;
    var ZeigerAufNeuenKnoten = document.createElement("OL");
    Liste.replaceNode(ZeigerAufNeuenKnoten);
    ZeigerAufNeuenKnoten.innerHTML = RetteInnerHTML;
  }
</SCRIPT>
<UL ID = "Liste" >
  <LI>Listeneintrag 1
  <LI>Listeneintrag 2
  <LI>Listeneintrag 3
  <LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Ersetze" onclick = "Ersetze()">
```

`.setAttribute()` Wert von vorhandenem Attribut setzen
 wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
 DOM wird nur bei Erzeugung geändert

Beispiel:

```
<HTML>
<HEAD>
```



```

<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachennamen definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitpunktUTC = Zeitpunkt.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitpunktUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

.setAttributeNode() Attribut einem Knoten zuweisen und Referenz liefern
 DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>

```



```

<SCRIPT>
function Hinzufuegen()
{
    // Attribut mit Wert erzeugen
    var ZeigerAufAttribut = document.createAttribute("title");
    ZeigerAufAttribut.value = "Tooltip-Text";

    // Attribut als Knoten erzeugen
    var AttributKnoten = ID_Div.setAttributeNode(ZeigerAufAttribut);
}
</SCRIPT>
</HEAD>
<BODY onload="Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

.setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert

In nachfolgenden Beispielen wurde aus Platzgründen die Zeichenkette von STYLE umgebrochen, was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
function width_init()
{
    ID_Span.style.setExpression("width",
                                " trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                "
                                "jsript"
                                );
}

function Berechne()
{alert(ID_Span.style.getExpression("width");}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
        STYLE="background-color:lightgreen;
                width:expression( trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                )
                "
    >
</SPAN>
<BUTTON onclick= Berechne();>
</BODY>
</HTML>

```

Beispiel 2:

```

ID_Span.style.setExpression("height","document.style.fontSize + 13");
ID_Span.style.setExpression("width","document.body.style.fontSize");
<SPAN ID="ID_Span"
    STYLE="background-color:lightgreen;
            width:expression( trueBlueSpan.style.pixelWidth
                            + oldYellowSpan.style.pixelWidth
                            )
            "
    >
</SPAN>

```

Beispiel 3 für Sekundenbalken:




```

<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

    function ZurueckSetzen()
    { Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"

```



```

        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON        ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P        STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

.swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

Beispiel:

```

<SCRIPT>
    function Tauschen()
    {Liste.children(0).swapNode(Liste.children(1)); }
</SCRIPT>
<UL ID = Liste>
    <LI>Listeneintrag 1
    <LI>Listeneintrag 2
    <LI>Listeneintrag 3
    <LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Tauschen" onclick = "Tauschen()">

```

4.3.1.3.3.2.4. **Collectionen zur Verwaltung des HTML-DOM im Internet Explorer**

Nachfolgende Collectionen sind nicht in allen Objekten implementiert (siehe einzelne Objekte).

Die Collectionen werden in ihren Eigenschaften und Methoden beschrieben. Aus Platzgründen sind Beispiele z.T. in den Anhang verlegt worden. Das gilt auch für identische Eigenschaften und Methoden der verschiedenen Collectionen.

4.3.1.3.3.2.4.1. **attributes Collection des HTML-DOM im Internet Explorer**

Diese Collection sammelt die Zeiger aller Attribute eines HTML-Elementes.

Hinweis: Es gibt noch das Objekt attribute, also ohne den Buchstaben S. Dieses Objekt dient zur Verwaltung von Attributen eines Objektes.

Die Collection attributes, also mit Buchstabe S, ist die Zeigersammlung von einzelnen attribute Objekten.

Syntax:

```

[ var FeldZeiger = ] object.attributes
[ var FeldElementZeiger = ] object.attributes[Index]

```

Index: Integer und ab 0
 muss in [] kodiert sein

Beispiel:

```

<SCRIPT>
    function ShowAttribs(ZeigerAufObjekt)
    {
        var ZeigerAufFeld = ZeigerAufObjekt.attributes;

        for (var Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld [Index];

            alert(
                ZeigerAufFeldElement.nodeName
                + '='
                + ZeigerAufFeldElement.nodeValue
                + '('
                + ZeigerAufFeldElement.specified
                + ')'
            );
        }
    }
</SCRIPT>

```



Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.getNamedItem() Zeiger auf Attribut liefern anhand des Attributnamen (analog zu ID oder NAME-Attribut)
ab IE 6.x

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        var ZeigerAufFeldElement = ZeigerAufFeld.getNamedItem("align");
        alert("ALIGN Attribut Wert = " + ZeigerAufFeldElement.value);
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P" ALIGN="center">Test</P>
</BODY>
</HTML>
```

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```
<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
    }
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>
```

.removeNamedItem() Attribut entfernen anhand Attributname (analog zu ID oder NAME-Attribut),
wobei danach der Standard-Attributwert automatisch weiterverwendet wird



(falls Standard vorhanden ist),
und Zeiger auf gelöscht Attribut liefern
ab IE 6

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function Entfernen()
    {
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.removeNamedItem("TITLE");
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV onclick="Entfernen();" ID="ID_Div" TITLE="Tooltip-Text ">
        Klick um den Tooltip-Text zu entfernen
    </DIV>
</BODY>
</HTML>
```

.setNamedItem() Attribut hinzufügen anhand Zeiger auf Attribut
wenn noch nicht im Feld vorhanden, so Anhängen an das Feldende
wenn schon im Feld vorhanden, so überschreiben und Referenz auf das
überschriebene Attribut (vor dem Überschreiben) liefern
Bsp: Attribut erzeugen document.createAttribute("title");
Hinweis: in HTML können Attributnamen groß oder klein geschrieben werden
ab IE 6

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function Hinzufuegen()
    {
        // Attribut mit Wert erzeugen
        var ZeigerAufAttribut = document.createAttribute("title");
        ZeigerAufAttribut.value = "Tooltip-Text ";

        // Attribut als Feldelement anhängen
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
    }
</SCRIPT>
</HEAD>
<BODY onload="Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>
```

4.3.1.3.3.2.4.2. *childNodes Collection des HTML-DOM im Internet Explorer*

Feld der Zeiger aller Kinder-Knoten eines Objektes, also Feld der Zeiger aller HTML-Knoten-Kinder **und** Textknoten-Kinder des Objektes
Collection dient zur Ermittlung **nur von Knoteneigenschaften laut DOM**, falls diese im Element implementiert sind:

Elementefolge NICHT laut HTML-Coding sondern laut DOM.

Element (Knoten) kann per HTML oder Methode .createElement() erzeugt worden sein (falls Methode erlaubt ist).

Diese Collection sammelt die Zeiger aller Kinder**knoten** eines HTML-Elementes.

Syntax:

```
[ var ZeigerAufFeld = ] object.childNodes
[ var ZeigerAufFeldElement = ] object.childNodes[Index]
```

object Zeiger auf Elternobjekt

Index Integer und ab 0
muss in [] kodiert sein

ZeigerAufFeldElement ist null, wenn Feldelement nicht vorhanden

Zugriff auf Element:

Je nach Art des Kind-Elementes stehen dem Kind Eigenschaften und Methoden zur Verfügung (siehe Objektbeschreibungen):

```
object.childNodes[Index].eigenschaft_des_kind
object.childNodes[Index].methode_des_kind
```

object Zeiger auf Elternobjekt



Index Integer und ab 0
muss in [] kodiert sein

Beispiel 1:

```
<SCRIPT>
    var ZeigerAufFeld = ID_Body.childNodes;
</SCRIPT>
<BODY ID="ID_Body">
    <SPAN >Test </SPAN>
</BODY>
```

Beispiel 2:

```
// DIV erzeugen
var ZeigerAufDivKnoten = document.createElement("DIV");

// B-Tag im DIV erzeugen
var ZeigerAufBKnoten = document.createElement("B");
ZeigerAufDivKnoten.insertBefore(ZeigerAufBKnoten);

// erst jetzt das DIV in den BODY einfügen, also DIV sichtbar machen
document.body.insertBefore(ZeigerAufDivKnoten);

// Collection referenzieren
var ZeigerAufFeld = ZeigerAufDivKnoten.childNodes;
```

Beispiel 3:

```
<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPTag = document.all.tags("P");
    if (ZeigerAufFeldAllerPTag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
        { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline"; } // auch .item(i) kodierbar
    }
</SCRIPT>
```

Beispiel 4:

```
<SCRIPT>
    var ErstesKind_Index = 0; // Index ab 0
    var ErstesKind_Name = Liste.childNodes(ErstesKind_Index).nodeName;
    alert(ErstesKind_Name); // liefert den Tagnamen 'LI' von Listenelement 1
                           // Hinweis: Listenelement 1 ist der Wert des Kindes
</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listenelement 2
        <LI> Listenelement 3
    </UL>
</BODY>
```

Beispiel 5:

```
<SCRIPT>
    function KnotenWertAendern( ZeigerAufListe,
                                IndexVonListenElement, // immer ab 0
                                Zeichenkette // Listenlement muss Text sein
                                )
    {
        var ReturnWert=false; // Annahme: Änderung schlägt fehl

        // prüfen auf UL-Tag
        if (ZeigerAufListe.nodeName == "UL")
        {
            // Anzahl der Listenelemente holen
            var AnzahlListenelemente= ZeigerAufListe.childNodes.length;
                                                    // immer ab 1

            // und Anzahl und Index prüfen
            if ( (AnzahlListenelemente > 0) // immer ab 1
                && (IndexVonListenElement >= 0) // immer ab 0
                && (IndexVonListenElement < AnzahlListenelemente)
                // Index ist zulässig zur Anzahl
            )
            {
                // hier würde die Änderung des Textes erfolgen
            }
        }
        ReturnWert = true;
    }

    // hier würde die Änderung des Textes erfolgen
    KnotenWertAendern(ZeigerAufListe, IndexVonListenElement, Zeichenkette);
```



```

    }
    {
        // Zeiger auf das Listenelement laut Index holen
        var ZeigerAufListenelement =
            ZeigerAufListe.childNodes(IndexVonListenelement);

        // existiert das Listenelement ?
        if (ZeigerAufListenelement)
        {
            // ZeigerAufListenelement ist nicht null

            // Listenelement ist Textelement ?
            if (ZeigerAufListenelement.nodeType == 3)
            {
                ZeigerAufListenelement.nodeValue =
                    Zeichenkette;

                ReturnWert =true;
            }
        }
    }

    return ReturnWert;
}
</SCRIPT>
<UL ID="Liste" onclick=" KnotenWertAendern(this, 0, 'Listenelement Neu')">
    <LI>Listenelement alt
</UL>

```

Beispiel 6:

```

<SCRIPT>
    function TextElementAendern()
    {
        var ZeigerAufTextElement = document.createTextNode("Neuer Text");
        var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
        ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
    }
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>

```

Beispiel 7:

```

<SCRIPT>
    function Anzeige()
    {
        var ZeigerAufOnClickEventQuelle=event.srcElement;
        var Feld =
            ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
        alert(
            "Anzahl LI : "
            + Feld.length
            + "\nErster Eintrag: "
            + Feld [0].childNodes[0].nodeValue
        );
    }
</SCRIPT>
<UL onclick="Anzeige()">
    <LI>Menuepunkt 1
    <UL>
        <LI> Menuepunkt 1.1
        <OL>
            <LI> Menuepunkt 1 1.1
            <LI> Menuepunkt 1 1.2
        </OL>
        <LI> Menuepunkt 1.2
        <LI> Menuepunkt 1.3
    </UL>
    <LI> Menuepunkt 2
    <UL>
        <LI> Menuepunkt 2.1
        <LI> Menuepunkt 2.3
    </UL>
</UL>

```



```

</UL>
<LI> Menüepunkt 3
</UL>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
 Attributnamen (analog zu ID oder NAME-Attribut) liefern
 außer bei Formular mit <INPUT TYPE=image ...>
 da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        { alert(ZeigerAufCollectionDocumentAll.item(i).tagName); }
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>

```

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
    var Text = "";

    if (ZeigerAufFeldAllerURN1 != null)
    {
        for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
        { Text += ZeigerAufFeldAllerURN1.item(i).id + ', '; }
        alert (Text);
    }
</SCRIPT>

```

4.3.1.3.3.2.4.3. children Collection des HTML-DOM im Internet Explorer

Feld der Zeiger aller **HTML-Elemente-Kinder** eines Objektes

Collection dient **auch** zur Ermittlung von Knoteneigenschaften, wenn diese im Element implementiert sind.

Elementefolge laut HTML-Coding und nicht laut DOM

Element kann per HTML oder Methode .createElement() erzeugt worden sein (falls Methode erlaubt ist).

Für das Objekt form.input image **muss** die children Collection verwendet werden.



Syntax:

```
[ var ZeigerAufFeld = ] object.children
[ var ZeigerAufFeldElement = ] object.children[Index [ , SubIndex] ]
```

object		Zeiger auf Elternobjekt
Index	oder	Integer ab 0 String Name oder ID des Elementes laut ID-Attribut bzw. NAME-Attribut muss in [] kodiert sein
SubIndex		optional Integer Unterindex also Unterelement eines Elementes nur kodieren wenn Index ein String ist
ZeigerAufFeldElement		ist null, wenn Feldelement nicht vorhanden

Beispiel 1:

```
<SCRIPT>
    var ZeigerAufFeld = ID_Body.children;
</SCRIPT>
<BODY ID="ID_Body">
    <SPAN>Test </SPAN>
</BODY>
```

Beispiel 2:

```
<SCRIPT>
    function Loeschen()
    {
        ID_Span.children[0].clearAttributes();
    }
</SCRIPT>
<SPAN ID="ID_Span">
    <DIV ID="ID_Div"
        ATTRIBUTE1="true"
        ATTRIBUTE2="true"
        onclick="alert('click');"
        onmouseover="this.style.color='#0000FF';"
        onmouseout="this.style.color='#000000';"
    >
        Test eines<B>Div</B>Elementes.
    </DIV>
</SPAN>
<INPUT TYPE="button" VALUE=" Loeschen" onclick="Loeschen()">
```

Beispiel 3:

```
<SCRIPT>
    function Hinzufuegen()
    {
        var ZeigerAufLI = document.createElement("LI");
        ID_Liste.children(0).insertAdjacentElement("beforeBegin", ZeigerAufLI);
        ZeigerAufLI.innerText = "Listeneintrag 0";
    }
</SCRIPT>
<BODY>
    <OL ID = "ID_Liste ">
        <LI>Listeneintrag 1</LI>
        <LI>Listeneintrag 2</LI>
        <LI>Listeneintrag 3</LI>
    </OL>
    <INPUT TYPE = "button" VALUE = " Hinzufuegen" onclick=" Hinzufuegen()">
</BODY>
```

Beispiel 4:

```
<SCRIPT>
    function Tauschen()
    {
        Liste.children(0).swapNode(Liste.children(1));
    }
</SCRIPT>
<UL ID = Liste>
    <LI>Listeneintrag 1
    <LI>Listeneintrag 2
```




```

        <LI>Listeneintrag 3
        <LI>Listeneintrag 4
    </UL>
    <INPUT TYPE = button VALUE = "Tauschen" onclick = "Tauschen()">

```

Beispiel 5:

```

<HEAD>
<SCRIPT>
    function Entfernen()
    {
        // versuche den Text zu entfernen
        try
        {
            var KindZeigerAufTextImDiv = ID_Div.children(0);
            ID_Div.removeChild(KindZeigerAufTextImDiv);
            // Achtung: Der Text ist noch sichtbar !!!!
        }
        // oder fange das Ereignis des bereits entfernten Textes ein
        // und behandle das Ereignis
        catch(x)
        {
            alert(        "Text wurde entfernt !\n"
                + "Das Dokument muss neu geladen werden !
            );
            document.location.reload();
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick="Entfernen()">
        Klick, um diesen Text zu entfernen !
    </DIV>
</BODY>

```

Beispiel 6:

```

<HEAD>
<SCRIPT>
    function Ersetze()
    {
        var KindZeigerAufDivText = ID_Div.children(0);
        var RetteInnerHTML = KindZeigerAufDivText.innerHTML;

        // prüfen auf Tag im Div-Text
        if (KindZeigerAufDivText.tagName=="B")
        {
            // Bold-Tag <B>gefunden, also I-Tag erzeugen
            var ZeigerAufNeuenSchriftStilTag =document.createElement("I");

            // komplettes ersetzen von Div-Text,
            ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
            ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
        }
        else
        {
            // keinen Bold-Tag <B>gefunden
            var ZeigerAufNeuenSchriftStilTag =document.createElement("B");

            // komplettes ersetzen von Div-Text,
            ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
            ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick="Ersetze()">
        Klicke für den Wechseln des <B>Schriftstils<B>
    </DIV>
</BODY>

```

Eigenschaften:



.length

Anzahl der Feldelemente also Feldlänge

Methoden:

.item()

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>

```

.tags()

Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPTag = document.all.tags("P");
    if (ZeigerAufFeldAllerPTag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
        { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
    }
</SCRIPT>

```

.urns()

Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
    var Text = "";

    if (ZeigerAufFeldAllerURN1 != null)
    {
        for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
        {Text += ZeigerAufFeldAllerURN1.item(i).id + ', ';}
        alert (Text);
    }

```



</SCRIPT>

4.3.1.3.3.2.4.4. tags Collection des HTML-DOM im Internet Explorer

Diese Collection sammelt die Zeiger aller HTML-Elemente, die gemeinsamen HTML-Tag besitzen. wird **nur** von Collectionen und Objekten instanziiert, die die Methode .tags() besitzen

z.B. childNodes Collection des DOM
 children Collection des DOM
 document.all Collection des DOM

Zugriff auf das Feld **nur** über die Methode .tags()

Syntax:

```
[ var Zeiger = ] zeiger_auf_collection_oder_objekt.tags(Kette)
```

Kette String mit HTML-Tag-Bezeichner

Zeiger weist auf ein leeres Feld, wenn keine HTML-Elemente mit dem Tag gefunden wurden

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
  var ZeigerAufFeldAllerPTag = document.all.tags("P");
  if (ZeigerAufFeldAllerPTag!=null)
  {
    for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
    { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
  }
</SCRIPT>
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

keine

4.3.1.3.3.2.5. command Objekt zum HTML-DOM des Internet Explorer

Das Objekt ist ein symbolisches Objekt zum Verwalten von externen vordefinierten Kommandos des IE. Diese Kommandos sind eine andere Variante von Objekterzeugungen (z.B. von HTML-Elementen) als Analogon zu Makros. Das Objekt besitzt nur Methoden, die an andere Objekte vererbt werden. Die meisten Methoden sind **erst** nach dem kompletten Laden des Dokumentes anwendbar. Falls die Methoden Werte liefern, so sind die Werttypen kommandospezifisch.

Erzeugung:

durch Browser

Syntax:

```
object.methode()
```

object Zeiger laut ID-Attribut

Beispiel 1:

```
<HTML>
<BODY>
  <H1 UNSELECTABLE="on">Demo</H1>
<SCRIPT>
  function AddLink()
  {
    var SelektierterText = document.selection.createRange();

    if (!SelektierterText == "")
    {
      // Link erzeugen
      document.execCommand("CreateLink");

      if (SelektierterText.parentElement().tagName == "A")
      {
        // markierten Text mit Eltern-Url ersetzen
        SelektierterText.parentElement().innerText=
          SelektierterText.parentElement().href;

        // Vordergrundfarbe setzen im Dokument
        document.execCommand(
          "ForeColor","false","#FF0033");
      }
    }
    else
    { alert("Bitte im blauen Text selektieren !");}
  }
</SCRIPT>
```



```

</SCRIPT>
<P UNSELECTABLE="on">
    Selektiere (markiere) im nachfolgenden blauen Text die Stelle mit
    dem Text MARKIERE_MICH.<BR>
    Danach auf das Button klicken.<BR>
    Anstelle von MARKIERE_MICH im blauen Text erscheint dort
    nun eine Url.
</P>
<P STYLE="color=#3366CC">
    Meine beliebteste Webseite MARKIERE_MICH bitte besuchen !
</P>
<BUTTON onclick="AddLink()" UNSELECTABLE="on">Klick</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function HandlerFuerOnMoveStart()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "green";
        ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
    }

    function HandlerFuerOnMove ()
    {
        ID_Span1.innerHTML = event.srcElement.offsetLeft;
        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandlerFuerOnMoveEnd()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    // 2-D Positionierung einschalten
    document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offserTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

Eigenschaften:

keine

Methoden:

.execCommand()

Kommando ausführen z.B. im aktuellen Dokument
in aktueller Selektion
im aktuellen Bereich
erst nach dem kompletten Laden des Dokumentes zulässig



Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
 Control = Element zur Steuerung analog zum HTML-Element (Tag)
 Input-Control = Element mit Eingabeeigenschaft

Syntax:

```
var Wert = object.execCommand(Command [, UserInterface] [, Value])
```

Command String als Kommando, wobei Gross-Kleinschreibung egal ist

"2D-Position"	absolute Positionierung von Elementen durch Bewegen im Dragging erlauben
"AbsolutePosition"	Element-Eigenschaft der Position auf "absolute" setzen nur für selektierbare Elemente nicht für STYLE-Deklarationen im Dokument
"BackColor"	lesen oder setzen der Hintergrundfarbe der aktuellen Selektion
"Bold"	Wechsel zwischen bold und nonbold in der aktuellen Selektion
"Copy"	Aktuelle Selektion in das Clipboard kopieren
"CreateBookmark"	Bookmark setzen oder lesen für aktuelle Selektion bzw. Einfügepunkt
"CreateLink"	Hyperlink in der aktuellen Selektion setzen oder einfügen bei Einfügen erscheint Dialog-Box
"Cut"	Aktuelle Selektion in das Clipboard verschieben
"Delete"	Aktuelle Selektion löschen Hinweis: Dokument nicht löscher
"FontName"	Font für aktuelle Selektion setzen oder holen
"FontSize"	Fontsize für aktuelle Selektion setzen oder holen
"ForeColor"	Vordergrundfarbe (Textfarbe) für aktuelle Selektion setzen oder holen
"FormatBlock"	Blockformat setzen
"Indent"	Ident des selektierten Textes erhöhen
"InsertButton"	in Textselektion das Button-Control einfügen, wenn eines bereits vorhanden so überschreiben
"InsertFieldset"	in Textselektion die Box einfügen, wenn eine bereits vorhanden so überschreiben
"InsertHorizontalRule"	in Textselektion die horizontale Linie einfügen wenn eine bereits vorhanden so überschreiben
"InsertIFrame"	in Textselektion den inline-frame (IFRAME) einfügen wenn einer bereits vorhanden so überschreiben
"InsertImage"	in Textselektion das Image einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputButton"	in Textselektion das Input-Button-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputCheckbox"	in Textselektion das Input-Check-Box-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputFileUpload"	in Textselektion das Input-File-Upload-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputHidden"	in Textselektion das Input-Hidden-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputImage"	in Textselektion das Input-Image-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputPassword"	in Textselektion das Input-Password-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputRadio"	in Textselektion das Input-Radio-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputReset"	in Textselektion das Input-Reset-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputSubmit"	in Textselektion das Input-Submit-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputText"	in Textselektion das Input-Text-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertMarquee"	in Textselektion das Marquee einfügen wenn eines bereits vorhanden so überschreiben
"InsertOrderedList"	Wechsel zwischen ordered list und normalen Block für aktuelle Textselektion
"InsertParagraph"	in Textselektion Zeilenumbruch einfügen wenn eines bereits vorhanden so überschreiben
"InsertSelectDropdown"	in Textselektion das Drop-Down-Selections-Control einfügen



		wenn eines bereits vorhanden so überschreiben
"InsertSelectListbox"	in Textselektion die List-Box-Selektion einfügen	wenn eines bereits vorhanden so überschreiben
"InsertTextArea"	in Textselektion das Input-Textarea-Control einfügen	wenn eines bereits vorhanden so überschreiben
"InsertUnorderedList"		
	Wechsel zwischen unordered list und normalen Block für aktuelle Textselektion	
"Italic"	Wechsel zwischen italic und non-italic für aktuelle Textselektion	
"JustifyCenter"	zentrieren für aktuelle Textselektion	
"JustifyLeft"	linksbündig für aktuelle Textselektion	
"JustifyRight"	rechtssbündig für aktuelle Textselektion	
"MultipleSelection"	Mehrfachselektion per CTRL+ bzw. Shift + erlauben	
"Outdent"	Outdent des selektierten Textes erniedrigen	
"OverWrite"	Wechsel zwischen überschreiben und nicht überschreiben	
"Paste"	Aktuelle Selektion aus Clipboard überschreiben	
"Print"	Druck-Dialogbox öffnen	
"Refresh"	aktuelles Dokument refreshen	
"RemoveFormat"	formatierende Tags der aktuellen Selektion entfernen	
"SaveAs"	Aktuelles Dokument speichern als Datei	
"SelectAll"	alles markieren (selektieren)	
"UnBookmark"	Alle Bookmark der aktuellen Selektion entfernen	
"Underline"	Wechsel zwischen underline und nicht-underline für aktuelle Textselektion	
"Unlink"	Alle Hyperlink der aktuellen Selektion entfernen	
"Unselect"	Alles demarkieren (de-selektieren)	
UserInterface	false	Default user interface soll nicht angezeigt werden (Dialogbox)
	true	user interface soll angezeigt werden (falls Kommando das unterstützt)
Value	z.B. String, number immer passend zu Command, kann optional sein Kodierung null (nicht numerisch Null !!) als Wert entspricht Weglassen des optionalen Wertes	

Kommando	IE ab	Dialogbox	Value
2D-Position	5.5	nein	true für on false für off
AbsolutePosition	5.5	nein	true für "absolut" false für nicht "absolut"
BackColor	4.x	nein	rrgbbb OHNE führendes # vordefinierter Farbname (browserspezifisch)
Bold	4.x	nein	null oder omit oder weglassen
Copy	4.x	nein	null oder omit oder weglassen
CreateBookmark	4.x	nein	String mit Ankername keine Leerkette
CreateLink	4.x	ja	String mit Url keine Leerkette
Cut	4.x	nein	null oder omit oder weglassen
Delete	4.x	nein	null oder omit oder weglassen
FontName	4.x	nein	String mit Fontname oder Fontliste Fontliste: Folge von Fonteigenschaften
FontSize	4.x	nein	String mit Fontsize von einschliesslich 1 bis einschliesslich 7
ForeColor	4.x	nein	rrgbbb OHNE führendes # vordefinierter Farbname (browserspezifisch)
FormatBlock	4.x	nein	String mit Block-Tag
Indent	4.x	nein	null oder omit oder weglassen
InsertButton	4.x	nein	String mit Attributen des Button-Control
InsertFieldset	4.x	nein	String mit Attributen der Box



InsertHorizontalRule	4.x	nein	String mit Attributen der Linie
InsertIFrame	4.x	nein	String mit Attributen des IFRAME
InsertImage	5.x	ja	String mit Pfad und Dateiname
InsertInputButton	4.x	nein	String mit Attributen des Input-Button-Control
InsertInputCheckbox	4.x	nein	String mit Attributen des Input-Checkbox-Control
InsertInputFileUpload	4.x	nein !!!	String mit Attributen des Input-Fileupload-Control
InsertInputHidden	4.x	nein	String mit Attributen des Input-Hidden-Control
InsertInputImage	4.x	nein	String mit Attributen des Input-Image-Control
InsertInputPassword	4.x	nein	String mit Attributen des Input-Password-Control
InsertInputRadio	4.x	nein	String mit Attributen des Input-Radio-Control
Kommando	IE ab	Dialogbox Value	
InsertInputReset	4.x	nein	String mit Attributen des Input-Reset-Control
InsertInputSubmit	4.x	nein	String mit Attributen des Input-Submit-Control
InsertInputText	4.x	nein	String mit Attributen des Input-Text-Control
InsertMarquee	4.x	nein	String mit Attributen des Marquee-Control
InsertOrderedList	4.x	nein	String mit Attributen des Ordered-List-Control
InsertParagraph	4.x	nein	String mit Attributen des Paragraph-Control
InsertSelectDropdown	4.x	nein	String mit Attributen des Drop-Down-Control
InsertSelectListbox	4.x	nein	String mit Attributen des List-Box-Control
InsertTextArea	4.x	nein	String mit Attributen des Text-Area-Control
InsertUnorderedList	4.x	nein	String mit Attributen des Unordered-List-Control
Italic	4.x	nein	null oder omit oder weglassen
JustifyCenter	4.x	nein	null oder omit oder weglassen
JustifyLeft	4.x	nein	null oder omit oder weglassen
JustifyRight	4.x	nein	null oder omit oder weglassen
MultipleSelection	5.5	nein	true für on false für off
Outdent	4.x	nein	null oder omit oder weglassen
OverWrite	4.x	nein	true für Überschreiben false für Nicht-Überschreiben
Paste	4.x	nein	null oder omit oder weglassen
Print	5.5	ja	null oder omit oder weglassen
Refresh	4.x	nein	kein String !!! null oder omit oder weglassen
RemoveFormat	4.x	nein	null oder omit oder weglassen
SaveAs	4.x	ja	wenn Dialogbox anzeigen so kann Wert null sein wenn Dialogbox nicht anzeigen so muss Wert die Paramter enthalten
SelectAll	4.x	nein	null oder omit oder weglassen
UnBookmark	4.x	nein	null oder omit oder weglassen
Underline	4.x	nein	null oder omit oder weglassen
Unlink	4.x	nein	null oder omit oder weglassen
Unselect	4.x	nein	null oder omit oder weglassen

Wert true wenn Kommando ausgeführt wurde
false wenn Kommando nicht ausgeführt wurde

.queryCommandEnabled() prüfen ob Kommando ausführbar ist
.queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht



.queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird
 .queryCommandValue() Wert eines Kommandos liefern

4.3.1.3.3.2.6. TextNode Objekt des HTML-DOM im Internet Explorer

Dieses Objekt ist ein symbolisches Objekt und

verwaltet Plain-Textdaten im DOM (keine HTML-Daten), also **Textknoten**

kann von Objekten instanziiert werden, die über nachfolgend beschriebene Methoden verfügen z.B. über

.createTextNode()

ist Analogon zur Stringverarbeitung

ist Basisobjekt für TextRange Objekt (siehe window.document.TextRange)

ab IE 6.x

Erzeugung:

durch Browser

Syntax:

object.methode()

object laut ID-Attribut

Textdaten erzeugen:

.createTextNode()

Plain-Textelement im Dokument erzeugen und Referenz liefern

Plaintext kann keine HTML-Tags enthalten

DOM wird geändert, da Dokument erweitert wird

Syntax:

[var Zeiger =] document.createTextNode([Text])

Text String Text als Inhalt des Elementes

Zeiger Referenz auf Objekt

Beispiel 1:

```
<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>
```

Beispiel 2: var TextKnoten = document.createTextNode("Test 1");

Textdaten ersetzen:

.replaceData()

Teilkette in einem Objekt ersetzen

Syntax:

object.replaceData(Offset, Anzahl, Kette)

Offset Integer

Startposition ab der ersetzt werden soll
ab 0

Anzahl Integer

Anzahl der ersetzenden Zeichen

ab 1

wenn Anzahl > object.length, so am Ende abgeschnitten

Kette

Zeichenkette, die ersetzt

liefert nichts

Beispiel:

```
var TextKnoten = document.createTextNode("Test 1");
TextKnoten.replaceData(5, 1, "2 "); // ergibt "Test 2"
```

Textdaten löschen:

.deleteData()

Teilkette aus einem Objekt entfernen

Syntax:

object.deleteData(Offset, Anzahl)

Offset Integer

Startposition der zu löschenden Teilkette
ab 0

Anzahl Integer

Anzahl der zu löschenden Zeichen

ab 1



wenn Anzahl > object.length, so kein Fehler

liefert nichts

Beispiel:

```
var TextKnoten = document.createTextNode("Test 1");
TextKnoten.deleteData(4, 2); // ergibt "Test"
```

Textdaten einfügen:

.insertData()

Teilkette in ein Objekt einfügen

Syntax:

object.insertData(Offset, Kette)

Offset Integer

Startposition ab der eingefügt werden soll
ab 0

Kette Zeichenkette

liefert nichts

Beispiel:

```
var TextKnoten = document.createTextNode("Test 1");
TextKnoten.insertData(4, "reihe"); // ergibt "Testreihe 1"
```

Textdaten anhängen:

.appendData()

String an das Ende des Objektes anhängen

Syntax:

object.appendData(Kette)

Kette Zeichenkette

liefert nichts

Beispiel:

```
var TextKnoten = document.createTextNode("Test 1");
TextKnoten.appendData("0"); // ergibt "Test 10"
```

4.3.2. window Objekt

Objekt eines Browserfenster

das im Browser erzeugt wird

das in browserspezifischen Versionen erzeugt werden kann

Das window Objekt hat diverse Zeiger auf andere Objekte:

z.B. beim IE und NS:

.document	Zeiger auf das Objekt document im Fenster
.event	Zeiger auf das Objekt event im Fenster
.history	Zeiger auf das Objekt history (Verlauf)
.location	Zeiger auf das Objekt location
.navigator	Zeiger auf das Objekt navigator

z.B. beim IE:

.screen	Zeiger auf das Objekt screen
---------	------------------------------

Diese Zeiger dienen **nur** der Zuordnung der Objekte zum Fenster. Solange es sich um das aktuelle Fenster handelt, kann also in der Punktnotation die Kodierung von window entfallen.

Das Objekt window beschreibt obige Objekt nicht: Aus Sicht des HTML-DOM sind Objekte, die im DOM hinterlegt sind, keine Kinder des Fensters, denn das ist im DOM nur **durch** das HTML-Dokument präsent.

Hinweis: Wenn ein Fenster mit dem Unterstrichsymbol rechts oben in der Fensterecke minimiert wurde, dann ist es durch Windows in die Hintergrund-Prioritätenfolge eingeordnet worden und arbeitet im Hintergrund bzw. garnicht. Mit anschließendem Maximieren (Symbol in der Fensterleiste rechts oben) wird das Fenster wieder angezeigt und das geladene Dokument **erneut** geöffnet. Konsequenz daraus ist, dass dieses Maximieren einem Neustart des Dokumentes entspricht, auch wenn der Programmierer diesen Zustand nicht erwünscht. Sound, der z.B. mit Start des Dokumentes erzeugt wird, erklingt erneut mit Maximierung nach einer Fensterminimierung. Analogon ist die Fenstereingung im Browser durch z.B. Ein- und Ausblenden der Favoritenleiste. Dieses Verhalten des Fensters mit seinem Dokument ist aber **nicht identisch** mit dem Verhalten nach einer Fenstergrößenänderung z.B. durch Rahmenverschiebung (resize). Die Aktionen des Fensters und seines Dokumentes bezüglich Resize müssen programmiert werden, sind also nicht standardmäßig vorhanden - im Gegensatz zum Verhalten mit/nach Fensterminimierung/-maximierung per Symbole in der rechten oberen Ecke der Fensterleiste. Sollte ein Frameset minimiert werden, dann wird das für den gesamten Frameset getan (Frameset hat 1 Fenster für alle Frames gesamt), allerdings mit Maximierung wird auch das Frameset-Dokument neu geladen. Wichtig dabei sind für Zeiger-Bezüge der Frames auf den Frameset per parent-Zeiger, dass die Daten im Frameset-Dokument mit Maximierung initialisiert werden und somit ebenfalls Daten der Frames, die im Frameset-Dokument abgelegt wurden, also z.B. Daten, die Verhaltensweisen der Frames vor einer Änderung der Frames-Inhalte gespeichert haben.

Erzeugung:

per Methode .open()



neues Fenster als unterstes Fenster in der Hierarchie instanzieren und öffnen, also anzeigen
mit öffnen wird Referenz auf das Fenster geliefert: Erstes Fenster wird beim Öffnen des Internet Explorer automatisch geöffnet.

Hinweis zu Scrollbalken: **Neu** erscheinende Scrollbalken verkleinern die Dimension des Browserfensters. Das ist zu beachten, wenn die Fensterdimension ohne bereits existierende Scrollbalken ermittelt wurde.

Beispiel: `var logischer_window_name;`

```
function oeffne_fenster()
{logischer_window_name=window.open();}

<A HREF="javascript:oeffne_fenster();">Fenster oeffnen</A>
<A HREF="javascript:logischer_window_name.close();">Fenster schliessen</A>
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
        + '<HEAD></HEAD>'
        + '<BODY >'
        + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
        + '<BR>'
        + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
        + ' onclick="opener.OnClickHandler();"'
        + '>'
        + '</BODY>'
        + '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

Zugriff:

auf Fenster mit Zeiger laut Erzeugung:

```
logischer_window_name.eigenschaft
logischer_window_name.methode()

logischer_window_name      laut open()
```

auf aktuelles Fenster:

```
window.eigenschaft
window.methode()

self.eigenschaft
self.methode()

eigenschaft
methode()
```

Hinweise: window ist synonym zu self

Innerhalb von Eventhandlern muss der korrekte Fensterbezug kodiert werden, egal ob es das aktuelle Fenster ist oder nicht. Folgende Kodierungen sind nicht zulässig:

```
eigenschaft
methode()
```

auf oberstes Fenster in der Fenster-Hierarchie:

```
top.eigenschaft
```




```
window.enableExternalCapture();
window.captureEvents(Event.CLICK | Event.MOUSEDOWN);
```

4.3.2.1.1. Eigenschaften

Falls es sich um das aktuelle Fenster handelt, so kann die Notation `window.eigenschaft` auf `eigenschaft` abgekürzt werden. Innerhalb von Eventhandlern ist immer **window.eigenschaft** zu kodieren, also die volle Referenzierung. Anstelle von `window` kann auch `self` kodiert werden.

Theoretisch ist auch `with (window) { }` kodierbar.

<code>._content</code>	Zeiger auf das Window-Objekt nur lesen nur NS 6.x
<code>.appCore</code>	nur NS 6.x
<code>.closed</code>	Zustand auf Geschlossenheit eines Fensters Syntax: [var Wert =] window.closed Wert false so ist Fenster offen true so ist Fenster geschlossen nur lesen
<code>.Components</code>	nur NS 6.x
<code>.controllers</code>	nur NS 6.x
<code>.crypto</code>	Zeiger auf das gleichnamige Objekt für Verschlüsselung anhand von Zertifikaten nur NS 6.x
<code>.defaultStatus</code>	Text der Statuszeile Verwendung in Eventhandler-Funktion: Es muss return true kodiert werden z.B. <code>onmouseover=" ...; return true;"</code> Syntax: window.defaultStatus [= Kette] [var Kette =] window.defaultStatus Kette String Plaintext lesen und schreiben
<code>.directories</code>	nur NS 6.x
<code>.document</code>	Zeiger auf das Objekt document im Fenster nur lesen
<code>.event</code>	Zeiger auf das Objekt event im Fenster nur lesen
<code>.frames</code>	Zeiger auf die Collection frames des Fensters nur lesen
<code>.history</code>	Zeiger auf das Objekt History nur lesen
<code>.innerHeight</code>	Höhe des Anzeigebereiches im Fenster in Pixel, ohne Leisten, Menü, Statuszeile etc. lesen und schreiben
<code>.innerWidth</code>	Breite des Anzeigebereiches im Fenster in Pixel, ohne Leisten, Menü, Statuszeile etc. lesen und schreiben
<code>.length</code>	Anzahl aller Frames bzw. IFrames im HTML-Dokument Integer, ab 0 nur lesen
<code>.location</code>	Zeiger auf das gleichnamige Objekt nur lesen
<code>.locationbar</code>	Zeiger auf die Adresszeile mit folgenden Eigenschaften: locationbar.visible liefert true wenn Adresszeile sichtbar liefert false wenn Adresszeile unsichtbar veränderbar per signiertem Script



.menubar	<p>Zeiger auf die Menüzeile mit folgenden Eigenschaften:</p> <p>.menubar.visible liefert true wenn Menüzeile im Fenster sichtbar liefert false wenn Menüzeile im Fenster unsichtbar</p> <p>veränderbar per signiertem Script</p>
.name	<p>physischer Fenster-Name eines Fensters laut open() entspricht Wert des Attributes TARGET eines Links</p> <p>Syntax:</p> <pre> window.name [= Kette] [var Kette =] window.name </pre> <p>Kette String</p> <p>lesen und schreiben</p> <p>Beispiel:</p> <pre> window.open("file.htm","Frame1"); window.name="MyWindow"; </pre>
.navigator	<p>Zeiger auf das Objekt navigator</p> <p>nur lesen</p>
.opener	<p>Zeiger auf dasjenige Fensters, das open() enthält also das das Kind-Fenster öffnet, welches in dieser .opener-Eigenschaft den Zeiger auf das Elternfenster enthält</p> <p>Bezug im geöffneten Fenster auf Instanzen des Aufrufers ist möglich</p> <p>Bezug des Aufrufers auf geöffnetes Fenster ist möglich</p> <p>Hinweis: Bei FRAME und IFRAME anstelle opener den Zeiger parent verwenden</p> <p>Öffnet ein Frame / IFrame ein Fenster, das damit nicht im Frameset läuft, so ist der Bezug vom Fenster aus auf das Frameset oder auf einen Frame im Frameset per window.opener.parent möglich, solange opener existiert.</p> <p>Syntax:</p> <pre> window.opener [= Zeiger] [Zeiger =] window.opener </pre> <p>lesen und schreiben</p>
.outerHeight	<p>Fensterhöhe in Pixel inkl. Anzeigebereich, Leisten, Menü, Statuszeile etc</p> <p>ohne signiertes Script minimal nur 100 Pixel</p>
.outerWidth	<p>Fensterbreite in Pixel inkl. Anzeigebereich, Leisten, Menü, Statuszeile etc.</p> <p>ohne signiertes Script minimal nur 100 Pixel</p>
.pageXOffset	<p>horizontale Position im gescrollten Dokument, die genau auf der linken oberen Ecke des Anzeigebereiches liegt, also Offset-Angabe gegenüber der linken obere Ecke des ungeschrollten Dokumentes, also gegenüber 0</p> <p>Hinweis: Dokument scrollbar im Anzeigebereich, wenn Größe des Dokumentes die des Anzeigebereiches überschreitet</p> <p>Offset ist 0, wenn Dokument noch nie gescrollt wurde</p>
.pageYOffset	<p>vertikale Position im gescrollten Dokument, die genau auf der linken oberen Ecke des Anzeigebereiches liegt, also Offset-Angabe gegenüber der linken obere Ecke des ungeschrollten Dokumentes, also gegenüber 0</p> <p>Hinweis: Dokument scrollbar im Anzeigebereich, wenn Größe des Dokumentes die des Anzeigebereiches überschreitet</p> <p>Offset ist 0, wenn Dokument noch nie gescrollt wurde</p>
.parent	<p>Referenz auf Elternfenster</p> <p>z.B. Zeiger auf das Fenster, das die FRAMESET-Deklaration enthält, oder das diesem Fenster übergeordnet ist</p> <p>muß nicht opener sein</p> <p>Test auf Existenz von parent per if (parent != self)</p> <p>Syntax:</p> <pre> [var Zeiger =] window.parent </pre> <p>nur lesen</p>
.personalbar	<p>ist selbst Objekt und enthält die Toolbar mit folgenden Eigenschaften:</p> <p>.visible ist true wenn Personal-Toolbar im Fenster sichtbar</p> <p>veränderbar per signiertem Script</p>
.pkcs11	nur NS 6.x
.prompter	nur NS 6.x



.screen	Zeiger auf das Objekt screen
.screenX	X-Koordinate links oben des Browser-Fensters 0 = linke obere Ecke des Bildschirms
.screenY	Y-Koordinate links oben des Browser-Fensters 0 = linke obere Ecke des Bildschirms
.scrollbars	Zeiger auf Objekt scrollbars, das die Scrollleisten mit folgenden Eigenschaften enthält: .visible ist true wenn Scrollbar im Fenster sichtbar veränderbar per signiertem Script
.self	Zeiger auf das aktuelle Fenster innerhalb der Fensterhierarchie bzw. Referenz auf aktuelles Fenster im FRAME ist synonym zu .window Syntax: self nur lesen
.sidebar	<p>Sidebar nur NS6.x Das Dokument, welches in die Sidebar geladen werden soll, muss folgende META-Tags besitzen:</p> <pre><HEAD> <META HTTP-EQUIV="expires" CONTENT="0"> <META HTTP-EQUIV="Pragma" CONTENT="no-cache"> </HEAD></pre> <p>Das Dokument, welches in die Sidebar geladen werden soll, kann einen Link z.B. per A-Tag besitzen, der im TARGET-Attribut den Wert "_content" besitzen muss: Bsp.: Dokument in der NS-Sidebar oeffnen</p> <p>Das Dokument, welches die Sidebar aufrufen will, kann dieses mit folgender Funktion tun:</p> <pre></HEAD> <SCRIPT LANGUAGE="JavaScript"> <!-- function DokumentInSideBar Laden(TitelKette, UrlKette) { // TitelKette String in " " bzw. ' ' mit freiem Wert // UrlKette String in " " bzw. ' ' mit absoluter Url-Angabe // prüfen ob Netscape-Objekt window.sidebar instanziert ist if (typeof window.sidebar == "object") { // prüfen ob Methode addPanel verfügbar ist if (typeof window.sidebar.addPanel == "function") // ohne () kodieren ! { // in der Sidebar neues Fenster öffnen und Dokument laden window.sidebar.addPanel(TitelKette, UrlKette, ""); } } } //--> </SCRIPT> </HEAD></pre> <p>enthält aktuellen Plain-Text der Statuszeile des Fensters (nicht Standardtext) auch schreibbar: wenn per Eventhandler, so muss dieser return true; enthalten</p>
.status	

Beispiel für Text buchstabenweise in Statuszeile anzeigen:

Der aktuelle Statuszeilentext wird als Teilkette von Position 0 bis zeichen_nr angezeigt, wobei zeichen_nr pro Anzeige um 1 erhöht wird.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
var statuszeile = new Array();
statuszeile[0] = "Text0";
statuszeile[1] = "Text1";
statuszeile[2] = "Text2";
```



```

var statuszeile_nr    = 0;
var zeichen_nr       = 0;

function textanzeigen ()
{
    if (position < statuszeile[statuszeile_nr].length) // Länge ab 1
    {
        // nächste Teilkette des aktuellen Statuszeilentextes anzeigen
        window.status = statuszeile[statuszeile_nr].substring(0, zeichen_nr);

        // nächste Position
        position++;
        setTimeout("textanzeigen()", 100);
    }
    else
    {
        // aktuelle Statuszeilentext komplett anzeigen
        window.status = statuszeile[statuszeile_nr];

        // nächste Statuszeile adressieren
        statuszeile_nr ++;
        if (statuszeile_nr >= statuszeile.length)
        { statuszeile_nr = 0; }

        // Position 0 einstellen
        zeichen_nr = 0;

        // Start der buchstabenweise Anzeige
        setTimeout("textanzeigen()", 1000);
    }
}

// -->
</SCRIPT>
</HEAD>
<BODY onLoad="textanzeigen()">
</BODY>
</HTML>

```

Beispiel für automatisch wechselnder Text in der Statuszeile:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var statuszeile = new Array();
    statuszeile[0] = "Text0";
    statuszeile[1] = "Text1";
    statuszeile[2] = "Text2";

    var statuszeile_nr = 0;

    function textanzeigen()
    {
        window.status = statuszeile[statuszeile_nr];
        statuszeile_nr ++;
        if (statuszeile_nr >= statuszeile.length) { statuszeile_nr = 0; }
        setTimeout("textanzeigen()", 1000);
    }

// -->
</SCRIPT>
</HEAD>
<BODY onLoad="textanzeigen()">
</BODY>
</HTML>

```

Beispiel für dauerhaftes Scrollen eines Textes in der Statuszeile:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">

```



```

<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext        = "";
    var zahler            = scrolltext_gesamt.length;

    function scrollen()
    {
        if (zahler == scrolltext_gesamt.length)
        {
            zahler = 0;
            scrolltext = scrolltext_gesamt;
        }
        else
        {
            zahler++; // 1 bis scrolltext_gesamt.length
            // Blank vorsetzen, also Kette um 1 Zeichen verlängern
            scrolltext = " "+scrolltext;

            // rausgerutschtes Zeichen abschneiden
            scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
            // Angaben ab 0
        }

        window.status = scrolltext;

        setTimeout("scrollen()", 100);
    }
// -->
</SCRIPT>
</HEAD>
<BODY onLoad="scrollen()">
</BODY>
</HTML>

```

Beispiel für einen Scrolltext in der Statusleiste, der angehalten wird, wenn Mauszeiger sich über einen Linkt bewegt:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext = "";
    var zahler = scrolltext_gesamt.length;
    var id;

    function scrollen()
    {
        if (zahler >= scrolltext_gesamt.length)
        {
            zahler = 0;
            scrolltext = scrolltext_gesamt;
        }
        else
        {
            zahler++; // 1 bis scrolltext_gesamt.length
            // Blank vorsetzen, also Kette um 1 Zeichen verlängern
            scrolltext = " "+scrolltext;

            // rausgerutschtes Zeichen abschneiden
            scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
            // Angaben ab 0
        }

        window.status = scrolltext;
        id=setTimeout("scrollen()", 100);
    }

    function scrollen_stoppen()
    {clearTimeout(id);}
// -->

```




```

</SCRIPT>
</HEAD>

<BODY onLoad="scrollen()">
    Bewegen Sie den Mauspfel &uuml;ber diesen
    <A    HREF="http://www.test.de"
        onMouseOver="scrollen_stoppen()"
        onMouseOut="scrollen()"
    >
    Link
    </A>
</BODY>
</HTML>

```

Beispiel für Anzeige eines Textes zu einem Hyperlink (HREF) für eine feste Zeitspanne in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var id;
    var anzeige_aktiv = false;
    var anzeige_zeit = 3000;      // 3000 Millisekunden = 3 Sekunden

    function anzeige_starten(href_text)
    {
        if(anzeige_aktiv)
        { clearTimeout(id); }

        window.status = href_text;

        id = setTimeout("anzeige_stoppen()", anzeige_zeit);

        anzeige_aktiv = true;

        return true;           // Wichtig !!!
    }

    function anzeige_stoppen ()
    {
        anzeige_aktiv = false;

        window.status = "";
    }
-->
</SCRIPT>
</HEAD>
<BODY>
    <A HREF= ... onMouseOver="javascript:return anzeige_starten('Hinweistext...')">
    ...
    </A>
</BODY>
</HTML>

```

Beispiel zur Anzeige eines Formularfeld-Hinweises in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    function hinweis_anzeigen(hinweis_text)
    { window.status = hinweis_text; }

    function hinweis_loeschen()
    { window.status = ""; }
-->
</SCRIPT>
</HEAD>
<BODY>
    <FORM>
        <INPUT TYPE=TEXT

```



```

        onFocus="hinweis_anzeigen('Hinweis');"
        onBlur="hinweis_loeschen();"
    >
</FORM>
</BODY>
</HTML>

```

Beispiel für blinkende Anzeige mit Zeitspanne von Text in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    var zeitspanne = 6000;        // 6 Sekunden blinken lassen, danach
                                //      Statuszeile löschen
    var anzeigezeit = 500;        // 0,5 Sekunden warten nach Setzen
                                //      bzw. Löschen der Statuszeile
    var id_blinken = null;        // muss auf null initialisiert sein !!

    function blinken_timer_loeschen
    {
        if (id_blinken != null)
        {
            clearTimeout(id_blinken); // blinken stoppen falls aktiv
            id_blinken = null;
        }
    }

    function blinken_stop()
    {
        blinken_timer_loeschen;
        window.status="";
    }

    function blinken_start(status_text)
    {
        blinken_timer_loeschen;
        blinken(true, status_text); // Blinken anstossen für
                                    // paralleles Arbeiten per Timer
        setTimeout("blinken_stop()",zeitspanne); // warten und danach blinken stoppen
    }

    function blinken(ein_aus, text)
    {
        if (ein_aus)
        {
            window.status = text;
            ein_aus=false; // im nächsten Aufruf die Statuszeile löschen
        }
        else
        {
            window.status = "";
            ein_aus=true; // im nächsten Aufruf die Statuszeile setzen
        }

        id_blinken = setTimeout("blinken(" + ein_aus + ")", anzeigezeit)
                                // nächsten Aufruf nach Ablauf der anzeigezeit starten
    }
-->
</SCRIPT>
</HEAD>
<BODY ... onLoad="blinken_start('Ich blinke !')">
</BODY>
</HTML>

```

.statusbar Zeiger auf Objekt statusbar, das die Statuszeile mit folgenden Eigenschaften enthält:
 .visible ist true wenn Statuszeile im Fenster sichtbar
 veränderbar per signiertem Script

.title String mit dem physischen Fensternamen, also den Text im Titel, also Titel des HTML-Dokumentes

.toolbar Zeiger auf das Objekt toolbar, das die Toolbar mit folgenden Eigenschaften enthält:
 .visible ist true wenn Toolbar im Fenster sichtbar



veränderbar per signiertem Script

.top Zeiger auf das oberste Fenster in der Fenster-Hierarchie
 Syntax:
 [var Zeiger =] window.top
 nur lesen

4.3.2.1.2. Methoden

Falls es sich um das aktuelle Fenster handelt, so kann die Notation window.methode() auf methode() abgekürzt werden. Innerhalb von Eventhandlern ist immer **window.methode()** zu kodieren, also die volle Referenzierung. Anstelle von window kann auch self kodiert werden.

Theoretisch ist auch with (window) { } kodierbar.

.alert() Dialogbox erzeugen
 1. Anzeige von:
 Ausrufungszeichen-Symbol
 variablen String
 OK-Button und
 2. warten auf Drücken des OK-Button
 Syntax:
 alert(String)
 String String oder Stringausdruck
 in " " bzw. " " kodieren
 Steuerzeichen wie "\n" zulässig
 sonst nur Plain-Text
 liefert nichts

.atob() liefert String, der das nach BASE64-Algorithmus kodierte String-Argument enthält
 Syntax:
 [var Kette2 =] atob(Kette1)
 Kette1 String
 zu kodierender Text
 Kette2 kodierter String

.back() lädt vorhergehendes HTML- Dokument laut History, wobei dabei Frames eines HTML-Dokumentes **nicht** beachtet werden
 Syntax:
 back()
 liefert nichts

.blur() Element den Focus wegnehmen und Event onblur auslösen
 Der Focus wird **nicht** automatisch auf irgend ein anderes Element gesetzt !
 Syntax:
 blur()
 liefert nichts

.btoa() liefert String, der das nach BASE64-Algorithmus kodierte String-Argument enthält
 Syntax:
 [var Kette2 =] btoa(Kette1)
 Kette1 String
 BASE64- kodierter Text
 Kette2 dekodierter String

.captureEvents() ordnet dem Window-Objekt Ereignisse zu, die anstelle der Standardbehandlung durch eine per Überschreiben des onXXX-Ereignis-Handlers gleichnamige Funktion (Überschreibung der Objektmethode) oder eine nicht-gleichnamige Funktion behandelt werden sollen
 Syntax:
 window. captureEvents(event_liste)
 event_liste: Folge von Eventbezeichnern
 Trennung durch | also logisches Oder
 Bsp. für Eventbezeichner

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS



onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER

liefert nichts

Hinweise zum Eventhandler:

- wenn das Ereignis nur einmal bearbeitet und danach wieder der Standardbehandlung zugeordnet werden soll, so muss die Funktion am Ende die Methode .releaseEvents() zum Ereignis aufrufen
- wenn das Ereignis nicht durch nachgelagerte Eventhandler in der Handlerhierarchie bearbeitet werden soll, so muss die Funktion mit return false; enden
- wenn das Ereignis durch nachgelagerte Eventhandler in der Handlerhierarchie bearbeitet werden soll, so muss die Funktion mit return true; enden

Beispiel:

```
function MouseOverHandler(){..}
window.onmouseover=MouseOverHandler;
// Achtung: nicht () kodieren, da die Funktion sonst
// sofort aktiviert wird !
window.captureEvents(Event.MOUSEDOWN);
```

.clearInterval()

stoppt einen Timer, der mit .setInterval() gestartet wurde

Syntax:

```
clearInterval(timer_id)
```

timer_id Integer
ist der Rückgabewert von .setInterval()

liefert nichts

.clearTimeout()

löscht ein Timeout, das mit .setTimeout() gesetzt wurde

Syntax:

```
clearTimeout(timeout_id)
```

timeout_id Integer
ist der Rückgabewert von .setTimeout()

liefert nichts

.close()

Fenster schliessen, das offen ist:

Beim Schliessen des ersten Fensters, das mit dem Starten des Browsers angezeigt wird, also beim Schliessen des Browsers, wird automatisch eine Dialogbox angezeigt.

Fenster muss nicht explizit mit open-Methode erzeugt worden sein

wenn Fenster mit .open() erzeugt wurde:

vor .close() ist der logische Windowname zu kodieren, der von .open() geliefert wurde, also logischer_window_name.close()

Fenster des Dokumentes schliessen: document.close()

innerhalb von Eventhandler immer **window.close()** anstelle von close() kodieren

Syntax:

```
.close()
logischer_window_name.close()
```

liefert nichts

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
function Test()
{window.close();}
</SCRIPT>
<BODY onclick="Test();">
Klick zum Auslösen von window.close()
</BODY>
```

Beispiel für Fenster schliesst sich selbst nach Wartezeit:

```
</HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function fenster_offnen_und_schliessen()
{

var fenster;
fenster=window.open("", "Fenster", "width=180,height=100");
```



```

        fenster.document.write("<H1>Ich schlieÙe mich nach 4 Sekunden</H1>");
        fenster.setTimeout('window.close()',4000); // 4 Sekunden
    }

    //-->
</SCRIPT>
</HEAD>
<BODY onload="fenster_offnen_und_schliessen()">
</BODY>
</HTML>

```

Beispiel:

```

function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
        + '<HEAD></HEAD>'
        + '<BODY >'
        + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
        + '<BR>'
        + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
        + ' onclick="opener.OnClickHandler();"'
        + '>'
        + '</BODY>'
        + '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();

```

.confirm()	<p>Dialogbox erzeugen mit</p> <ol style="list-style-type: none"> 1. Anzeige Fragezeichen, String, OK/Abbrechen-Button 2. warten auf Drücken eines der beiden Button <p>Syntax:</p> <pre>[var Wert =] confirm(Kette)</pre> <p>Kette String</p> <p>Wert true für OK false für Abbrechen</p>
.disableExternalCapture()	<p>schaltet Wirkung von .enableExternalCapture() ab Ereignissüberwachung in einem Fenster abschalten Beispiel für Anwendung: Fremdseite in einem Frame kann die Seite nicht überwachen, die die Fremdseite im Frame anzeigt</p> <p>Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !</p> <p>Syntax:</p> <pre>disableExternalCapture()</pre> <p>liefert nichts</p>
.dump()	<p>nur NS 6.x</p>
.enableExternalCapture()	<p>aktiviert die Kontrolle einer Fremdseite, die in einem Frame dargestellt wird, über die Seite, die den Frame inne hat Ereignissüberwachung in einem Fenster einschalten Beispiel für Anwendung: Fremdseite im Frame kann Seite überwachen Überwachung per .captureEvents() aktivierbar</p> <p>Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !</p> <p>nur mit signiertem Script</p> <p>Syntax:</p> <pre>enableExternalCapture()</pre>



	liefert nichts
.find()	Suche im Dokument nach Zeichenkette Syntax: [var Wert =] find([zeichenkette],gross_klein,such_richtung) zeichenkette wenn nicht kodiert, so automatisch Suchfenster geöffnet gross_klein wenn true, so Unterscheidung von Gross und Klein such_richtung wenn true, so rückwärts suchen Wert true, sobald zeichenkette gefunden
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes anwendbar Syntax: object.focus() liefert nichts
.forward()	lädt nachfolgendes HTML- Dokument laut History, wobei dabei Frames eines HTML-Dokumentes nicht beachtet werden Syntax: forward() liefert nichts
.getAttention()	nur NS 6.x
.getSelection()	aktuelle Selektion referenzieren nur NS 6.x

Beispiel: Suchmaschine einbinden

```
var markierter_text=document.getSelection();
// oder var markierter_text=prompt('Suchbegriff: ');
var suchmaschinen_url='http:// .....';
var suchmaschinen_parameter='.....';

if (markierter_text)
{location.HREF=suchmaschinen_url + suchmaschinen_parameter + escape(markierter_text);}
else
{location.HREF=suchmaschine_url; }
```

Beispiel für altavista: suchmaschinen_url 'http://altavista.de/'
suchmaschinen_parameter 'cgi-bin/query?pg=q&what=web&q='

Hinweis: escape() setzt Umlaute und Leerzeichen in korrekte Darstellung um

.handleEvent()	Eventhandler aufrufen, der dem Ereignis laut Parameter event_objekt gerade zugeordnet ist diese Methode wird also innerhalb des Programmcodes eines Eventhandlers verwendet, der das Ereignis von einem anderen Eventhandler als Argument bekommen hat verwendet für eine andere Eventverarbeitungs-Hierarchie als die Standardhierarchie Syntax: handleEvent(event_objekt_zeiger) liefert nichts
.home()	lädt Startseite laut Browsereinstellung, also nicht die Startseite des aktuellen HTML-Dokument-Projektes Syntax: home() liefert nichts
.moveBy(x,y)	Fenster verschieben um Pixeldifferenz Fenster ganz aus dem BS schieben ist eventuell möglich Syntax: moveBy(iX, iY) iX Integer, X-Koordinatendifferenz, auch negativ iY Integer, Y-Koordinatendifferenz, auch negativ liefert nichts

Beispiel für Fenster des Browser rausschieben und danach neu anzeigen:

<HTML>



```

<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    var BrowserFenster_AktuelleBreite=2000; // sollte größer als max. übliche Auflösung sein
    var BrowserFenster_AktuelleHoehe=2000; // sollte größer als max. übliche Auflösung sein

    function moveWin()
    {
        for (var i = 1; i < BrowserFenster_AktuelleBreite; i++)
        {window.moveBy(1, 1);}

        window.moveBy((-1)* BrowserFenster_AktuelleBreite,(-1) * BrowserFenster_AktuelleHoehe);
    }
//-->
</SCRIPT>
</HEAD>
<BODY onload="moveWin();">
</BODY>
</HTML>

```

.moveTo() Fenster verschieben auf Pixelpos bezüglich linke obere Ecke des Screen
(Ursprung (0,0) liegt in der linken oberen Ecke)
Fenster ganz aus dem BS schieben ist eventuell möglich
Syntax:
 moveTo(x, y)

 x horizontale absolute Position in Pixel der linken oberen Fensterecke, Integer >=0
 x vertikale absolute Position in Pixel der linken oberen Fensterecke, Integer >=0

liefert nichts

.open() neues Fenster erzeugen als unterstes der aktuellen Fensterhierarchie
und dann oberstes sichtbares Fenster öffnen (anzeigen, rendern)
Fenster mit kodiertem .open() ist der .opener
Syntax:
 [var ZeigerAufWindow =] open([URL] [, Name] [, Features] [, Replace])

innerhalb von Eventhandler immer **window.open()** anstelle von open() kodieren

URL String mit Url des Dokumentes, das nach dem Öffnen in das Fenster geladen wird
kann Leerkette sein
Daten per '?' + escape() **nicht** übergebbar

Name String
physische Window-Name: identisch mit Wert laut Attribut
TARGET z.B. im Link
dient der Referenz
auch 'null' kodierbar

Features String als Parameterliste mit Kommatrennung
(Liste der Fensterkomponenten)
gesamte Liste in " " bzw. ' ' setzen
z.B. "fullscreen=yes, toolbar=yes"
gesamte Liste ist 1 Zeichenkette,
die in 1 Quelltextzeile passen muss,
oder in Teilketten zerlegt mit dem
+ Operator zusammengesetzt wird
Blanks in Liste **nicht** zulässig
Listenelement: option=wert
wenn mindestens 1 Element kodiert, so alle
anderen nicht kodierten Elemente
automatisch deaktiviert, also immer alle
gewünschten Optionen kodieren !!!
Standardwert eines Merkmals, wenn
Liste kodiert wurde: no oder 0
Liste nicht kodiert wurde: yes oder 1
keine Standardwerte vorhanden für Pixelangaben
da diese vom Browser automatisch
belegbar sind

alwaysLowered = { yes | no }



yes: Fenster öffnen und permanent als
unterstes in der Fensterhierarchie
belassen
nur per signiertem Script
beachte .setZOptions()

alwaysRaised = { yes | no }
yes: Fenster öffnen und permanent als
oberstes in der Fensterhierarchie belassen
nur per signiertem Script
beachte .setZOptions()

dependent = { yes | no }
yes: Fenster an den .opener bezüglich
Fensterschliessen koppeln: auch
schliessen, wenn .opener geschlossen
wird
sowie geöffnetes Fenster nicht in
der Taskleiste von Windows
anzeigen

directories = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes oder 1 Directory Buttons anzeigen

height = Pixelwert, Fensterhöhe
nur mit signiertem Script < 100
ohne signiertem Script: wenn < 100,
so auf 100 automatisch gesetzt

hotkeys = { yes | no }
yes: alle sicherheitsrelevanten Tasten
verwendbar machen
no: nur noch ALT+F4 funktioniert
(schliessen des Fensters)

innerHeight= anzeigebereich_höhe_in_pixel
ohne signiertes Script: >= 100
Anzeigebereich = Fenster abzüglich
Leisten, Scrollbars etc.

innerWidth= anzeigebereich_breite_in_pixel
ohne signiertes Script: >= 100
Anzeigebereich = Fenster abzüglich
Leisten, Scrollbars etc.

location = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw 1 für URL-Eingabezeile anzeigen
(Adresszeile anzeigen)

menubar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Menubar anzeigen
(Leiste der Pull-DownMenüs anzeigen)

outerHeight= fenster_höhe_in_pixel
ohne signiertes Script: >= 100
Fenster = Anzeigebereich sowie Leisten,
Scrollbars etc.

outerWidth= fenster_breite_in_pixel
ohne signiertes Script: >= 100
Fenster = Anzeigebereich sowie Leisten,
Scrollbars etc.

personalbar = { yes | no }
yes: persönliche Toolbar anzeigen

resizable = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Größenveränderung des



Fensters erlaubt per Mausziehen
beachte .setResizable()

screenX= horizontale_pixel_pos des Fensters
bezüglich dem Bildschirm,
dessen Ursprung (0,0) in der
linken oberen Ecke liegt

screenY= vertikale_pixel_pos des Fensters
bezüglich dem Bildschirm,
dessen Ursprung (0,0) in der
linken oberen Ecke liegt

scrollbars = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Scrollbalken anzeigen
sobald Fensterinhalt größer als
Anzeigebereich des Fensters

status = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Statuszeile anzeigen

titlebar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Titelzeile anzeigen
Titel werden immer angezeigt bei Dialog-Box
nur mit signiertem Script setzbar

toolbar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Toolbar (Navigationsleiste)
anzeigen (Button Zurück etc.)

width = Pixelwert, Fensterbreite,
nur mit signiertem Script <= 100
ohne signiertem Script: wenn < 100,
so auf 100 automatisch gesetzt

z-lock = { yes | no }
yes: Fenster kann kein anderes
überlagern
nur per signiertem Script
beachte .setZOptions()

Replace true, so History-Eintrag des Dokumentes, in dem das neue
Fenster erzeugt wird, ersetzen durch den Eintrag
des neuen Fensters, also Zurück zum Dokument,
dass das Fenster öffnet, nicht möglich
false, so neuen History-Eintrag zum neuen Fenster
erzeugen, also zurück zum alten Fenster möglich

ZeigerAufWindow

Referenz (ID, logischer Windowname) auf das neue
Fenster z.B. für close-Methode

Beispiel window.open("Sample.htm",null,"height=200,width=400,status=yes,toolbar=no,menubar=no,location=no");

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}
```

....

// Fenster öffnen

```
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;
```

```
var Kette= '<HTML>'
```



```
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ 'onclick="opener.OnClickHandler();"'
+ '>'
+ '</BODY>'
+ '</HTML>';
```

```
FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.print() ruft Dialog-Box-Druckerfenster auf zum Druck des Dokumentes im aktuellen Fenster (Frame)
entspricht Druckbutton bzw. Datei-Menü-Drucken
löst folgende Ereignisse aus:
 onbeforeprint
 onafterprint

Syntax:
 print()
liefert nichts

Beispiel für Verwendung der Ereignisse per Handler

```
onbeforeprint     Handler blendet Teile der Seite ein/aus, die gedruckt werden sollen
onafterprint     Handler hebt Veränderungen von onbeforeprint wieder auf
```

Beispiel für Ausdruck des aktuellen Dokumentes:

```
<BODY>
<INPUT TYPE="button" VALUE="Drucken" onclick="javascript:self.print()">
</BODY>
```

.prompt() Eingabefenster erzeugen, Meldungstext anzeigen, Eingabezeile vorbelegen und anzeigen, OK- bzw. CANCEL-Button anzeigen und auf Eingabe in die Zeile und anschliessendem Druck auf OK warten

Syntax:
 [var Kette =] prompt("meldungstext" [, "vorgabe_text"])

meldungstext	String, frei wählbar
vorgabe_text	String, frei wählbar wenn nicht kodiert, so "undefined" als Standard
Kette	Zeichenketten-Ergebnis der Eingabe

Beispiel: var eingabe = prompt(.....);

.releaseEvents() ordnet dem Window-Objekt Ereignisse zu, die durch die Standardbehandlung bearbeitet werden
hebt zugleich ein aktuelles .captureEvents() zum Ereignis auf

Syntax:

```
releaseEvents(event_liste)
```

event_liste: Folge von Eventbezeichnern

Bsp. für Eventbezeichner

Trennung durch | also logisches Oder

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS
onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER

liefert nichts

.resizeBy() Fenstergröße um Pixeldifferenz verändern
funktioniert nur, wenn open-Merkmal resizable=yes kodiert wurde

Syntax:

```
resizeBy(x,y)
```



x ist horizontale Spanne in Pixel , Integer, auch negativ
 y ist vertikale Spanne in Pixel , Integer, auch negativ
 Fenstergröße auf weniger als 100 x 100 nur mit signiertem Script
 liefert nichts

.resizeTo()

Fenstergröße neu dimensionieren
 Syntax:

resizeTo(x, y)

x ist Breite in Pixel, Integer, >0
 y ist Höhe in Pixel, Integer, > 0
 Fenstergröße auf weniger als 100 x 100 nur mit signiertem Script
 liefert nichts

Beispiel für Fensterauflösung ändern:

javascript:window.resizeTo(**640,480**); javascript:window.resizeTo(**800,600**); javascript:window.resizeTo(**1024,768**)

Beispiel für sich aufblasendes Fenster von 100x100 bis auf 640x480

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    var start_hoehe=100;
    var start_breite=100;
    var max_hoehe=480;
    var max_breite=640;
    var aktuelle_hoehe=start_hoehe;
    var aktuelle_breite=start_breite;
    var y=5;
    var TimerID=null;
    var fenster;

function start()
{
    fenster=window.open("", "", "scrollbars");

    if ( document.layers || document.all)
    {
        fenster.resizeTo(start_breite,start_hoehe);
        fenster.moveTo(0,0);
        blasen();
    }
    else
    {alert("Weder Netscape noch Microsoft erkannt !");}
}

function blasen()
{
    if (aktuelle_hoehe>=max_hoehe)
    {x=0;}

    fenster.resizeBy(5,y);
    aktuelle_hoehe+=5;
    aktuelle_breite+=5;

    if (aktuelle_breite>=max_breite)
    {
        alert("Maximal aufgeblasen !");
        fenster.close();
        x=5;
        TimerID=null;
    }
    else
    {TimerID=setTimeout("blasen()",50);}
}
//-->
</SCRIPT>
</HEAD>
```



```

<BODY>
<A      HREF="javascript:start()"
onMouseOver="javascript:window.status='Oeffne Fenster';return true;" // Text nach Statuszeile
onMouseout="javascript:window.status=";" // Statuszeile löschen
>Oeffne NEUES Fenster mit 100x100 Pixel und blase es auf 640x480 Pixel !
</A>
</BODY>
</HTML>

```

`.routeEvent()` aktiviert das Weiterreichen des Events zum in der Eventhierarchie untergeordneten Eventhandler standardgemäß landen die Events immer beim Window-Objekt und werden dort behandelt, also nicht nach unten durchgereicht

Syntax:

```
routeEvent(ereignis_objekt_zeiger)
```

liefert nichts

Beispiel

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript">
<!--
function MouseDownFuerInputButton()
{ ... }

function OnMouseDownEventWeiterreichen(event)
{window.routeEvent(event);}

window.onmousedown= OnMouseDownEventWeiterreichen;
// Achtung: ohne () kodieren, sonst wird Funktion sofort ausgeführt
window.captureEvents(Event.MOUSEDOWN);

// -->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
<INPUT TYPE=button
VALUE="Weiterreichen"
onmousedown= "MouseDownFuerInputButton();"
>
</FORM>
</BODY>
</HTML>

```

Ablauf:

Ablauf im Script-Teil:

Der Script-Teil im Head wird zuerst abgearbeitet, also gilt:

Mousedown-Event durch `OnMouseDownEventWeiterreichen` bearbeiten lassen und nicht durch Standardbehandlung, also Ereignis weiterunter leitbar.

Hinweis: Standardbehandlung erfolgt durch das Window-Objekt, das Ereignisse nicht weiter runter leitet.

Ablauf im Body-Teil:

Es wird auf das Input-Button gedrückt.

Wegen `captureEvents()` wird das Mausereignis ZUERST von `OnMouseDownEventWeiterreichen` bearbeitet. Damit wird auf das Weiterleiten aktiviert.

Das untergeordnete Element ist der Input-Button. Damit wird `MouseDownFuerInputButton` aktiviert.

Hinweis: Dieses Beispiel hinkt, da die Standardbehandlung letztendlich das gleiche bewirkt aber auf anderen Wegen.

`.scroll()` deprecated und zu ersetzen durch `scrollBy` bzw. `.scrollTo()`

linke obere Ecke des Dokumentes verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters, also scrollen

bewirkt Veränderung von `.pageXOffset` und `.pageYOffset`

Hinweis: wenn Dokumentdimension > Anzeigebereichdimension, so ist Scrollen nötig

Syntax:

```
scroll(x, y)
```

x	horizontal scroll Offset in Pixels, Integer, > 0
y	vertikal scroll Offset in Pixels, Integer, > 0

liefert nichts

`.scrollBy()` linke obere Ecke des Dokumentes verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters, also scrollen



ist anstelle von `.scroll()` zu verwenden
 bewirkt Veränderung von `.pageXOffset` und `.pageYOffset`
 Hinweis: wenn Dokumentdimension > Anzeigebereichdimension, so ist Scrollen nötig
 Syntax:

```
scrollBy(x, y)
```

x horizontal scroll Offset in Pixels
 Integer
 > 0 so Verschiebung nach rechts
 < 0 so Verschiebung nach links

y vertikal scroll Offset in Pixels
 Integer
 > 0 so Verschiebung nach unten
 < 0 so Verschiebung nach oben

liefert nichts

`.scrollByLines()` nur NS 6.x

`.scrollByPages()` nur NS 6.x

`.scrollTo()` linke obere Ecke des Dokumentes verschieben auf neue Pixelposition bezüglich linker oberer Ecke des Fensters, also scrollen
 ist anstelle von `.scroll()` zu verwenden
 bewirkt Veränderung von `.pageXOffset` und `.pageYOffset`
 Syntax:

```
scrollTo(x, y)
```

x horizontale Position in Pixel
 Integer, >=0

y vertikale Position in Pixel
 Integer, >=0

liefert nichts

`.scrollX` nur NS 6.x

`.scrollY` nur NS 6.x

`.setActive()` Objekt für die Eventdurchreichung aktivieren, aber ohne es zu fokussieren und ohne es scrollbar zu machen

Syntax:

```
setActive()
```

liefert nichts

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
  var ID_Fenster;

  function FensterErzeugen()
  {
    ID_Fenster = window.open( "test.htm",
                              " ID_Fenster",
                              "top=10px,left=480px,height=375px,width=200px,resizable=1"
                              );
    this.focus();
  }

  function ButtonAktivieren()
  {window.parent.ID_Fenster.ID_Button.setActive();}
</SCRIPT>
</HEAD>
<BODY onload="FensterErzeugen();">
  <BUTTON ID="ID_Button" onclick="ButtonAktivieren();">
    Button aktivieren
  </BUTTON>
</BODY>
</HTML>
```

`.setCursor()` nur NS 6.x



<code>.setHotKeys()</code>	Hotkeys des Fensters (außer im Menü) aktivieren bzw. deaktivieren Standard: immer aktiviert Syntax: <code>setHotKeys(Wert)</code> Wert true, so Hotkeys aktiviert false, so Hotkeys deaktiviert liefert nichts
<code>.setInterval()</code>	endlos-periodischer Aufruf eines Codes mit jeweiligem vorherigen Warten in Millisekunden (Timer) (getimte Rekursion) Der mit <code>setInterval()</code> aufgerufene Code wird zyklisch aktiviert, wobei nach dem ERSTEN Aufruf von <code>setInterval()</code> mit der Folgeanweisung hinter <code>.setInterval()</code> sofort weitergemacht wird, während die Rekursion parallel läuft. Damit gilt: <code>setInterval()</code> kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da sonst diese Anweisungen während der parallelen rekursiven Ereignisüberwachung bereits abgearbeitet werden. Syntax: <code>[var TimerID =] setInterval(Code, MilliSeconds [, Language])</code> Code auszuführender Code bzw. Zeiger auf Codebaustein Zeiger empfehlenswert, wenn Code in externer Datei liegt String empfehlenswert, wenn Code im aktuellen Dokument liegt Übergabe von Argumenten möglich MilliSeconds Integer, Wartezeit, erst nach deren Ablauf wird Code aktiviert Language String als Sprache des Codes (analog zum LANGUAGE-Attribut) z.B. "JavaScript". TimerID Integer ID für den Stopp der periodischen Aufruffolge mit der Methode <code>.clearInterval()</code>

Beispiel 1

```
String      window.setInterval("someFunction()", 5000);
Zeiger      window.setInterval(someFunction, 5000);
```

Beispiel 2

```
function callback1(){alert("callback1");}

function callback2(){alert("callback2");}

function chooseCallback(Nummer)
{
    switch (Nummer)
    {
        case 0: return callback1;
        case 1: return callback2;
        default: return "";
    }
}

var Nummer = 1;
window.setInterval(chooseCallback(Nummer), 5000);
```

Beispiel 3

```
var SekundenZahler=0;
var timer_id=null;

function ZyklischeAktion()
{
    SekundenZahler++;
    window.status= SekundenZahler + " Sekunden ";

    if (      ( SekundenZahler >= 60 )
```



```

        && ( timer_id != null)
    )
    {window.clearInterval(timer_id);}
}

timer_id=window.setInterval(ZyklischeAktion,1000);

```

Beispiel 4

```

var SekundenZahler=0;
var timer_id=window.setInterval("window.status= SekundenZahler++",1000);

```

.setResizable()

Veränderbarkeit der Fenstergröße aktivieren/ deaktivieren

Syntax:

setResizable(Wert)

Wert true für Veränderbarkeit aktiv

false für Veränderbarkeit nicht möglich

liefert nichts

.setTimeout()einmaliger und somit **nicht periodischer** Aufruf eines Codes mit einmaligem vorherigen Warten in Millisekunden (Timer)

Der mit setTimeout() aufgerufene Code wird **nicht zyklisch** aktiviert, wobei nach dem Aufruf von setTimeout() mit der Folgeanweisung hinter setTimeout() sofort weitergemacht wird. Damit gilt: setTimeout() kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da diese Anweisungen bereits während der Ereignisüberwachung abgearbeitet werden.

Syntax:

[var TimerID =] setTimeout(Code, MilliSeconds [, Language])

Code

auszuführender Code bzw. Zeiger auf Codebaustein
 Zeiger empfehlenswert, wenn Code in externer Datei liegt
 String empfehlenswert, wenn Code im aktuellen Dokument liegt
 Übergabe von Argumenten möglich

MilliSeconds

Integer, Wartezeit,
 erst nach deren Ablauf wird Code aktiviert

Language

String als Sprache des Codes
 (analog zum LANGUAGE-Attribut)
 z.B. "JavaScript"

TimerID

Integer
 ID für den Stopp der periodischen Aufruffolge mit der Methode .clearTimeout()
 wird nur benötigt, wenn der Code auf eine rekursive Funktion weist, wobei der Zyklusabbruch durch .clearTimeout() erfolgen muss
 Hinweis: ist keine Rekursion nötig, soll aber zyklisch aufgerufen werden, dann .setInterval() verwenden

Beispiel 1

```

window.setTimeout("alert('Hallo')", 1000);

```

Beispiel 2

```

var Text = "Hallo ";
window.setTimeout( "alert("
                    + Text
                    + ")",
                    1000
                );

```

Beispiel 3

```

<SCRIPT>
function Start(ZeigerAufObjekt)
{window.setTimeout("Kode(" + ZeigerAufObjekt.id + ")", 3000);}

function Kode(ID)
{
    var Zeiger = eval(ID);
    Zeiger.style.display="none";
}

```



```

</SCRIPT>
<INPUT TYPE=button VALUE="Unsichtbar in 3 Sekunden"
      ID="ID_Button" onclick=" Start(this)"

```

Beispiel 4

```

var SekundenZahler=0;
var timeout_id=null;

function ZyklischeAktion()
{
    SekundenZahler++;
    window.status= SekundenZahler + " Sekunden ";

    if (      ( SekundenZahler >= 60 )
        && ( timeou_id != null)
        )
    {window.clearTimeout(timeout_id);}

}

timeout_id=window.setTimeout(ZyklischeAktion,1000);

```

Beispiel 5

```

var timeout_id=window.setTimeout("window.status= 'Es ist 1 Sekunde vergangen !'",1000);

```

`.setZOptions()` Art und Weise der Fensterüberlagerung unabhängig vom Aktivzustand des Fensters definieren
 Syntax:
 `setZOptions(zustand)`

 zustand: Zeichenkette mit folgenden Werten
 "alwaysRaised" Fenster immer oberstes auch wenn nicht aktiv
 "alwaysLowered" Fenster immer unterstes auch wenn aktiv
 "z-lock" Fenster bleibt in aktueller Überlagerungssituation
 egal ob aktiv oder nicht

 liefert nichts

`.sizeToContent ()` nur NS 6.x

`.stop()` stoppt das Laden des Dokumentes in das Fenster
 Syntax:
 `stop()`
 liefert nichts

`.updateCommands()` nur NS 6.x

4.3.2.1.3. **window.document** Objekt des Netscape

Erzeugung:
 durch den Browser

Das HTML-Dokument

umfasst sämtlichen Code **zwischen** <HTML...> und </HTML>
 ist der Container für die Elemente der Webseite z.B. BODY oder DIV
 Code hinter </HTML> wird nicht geparkt aber eventuell als Plain-Text angezeigt
 Ein Dokument muss zwar in ein Fenster geladen werden, aber das Dokument wird nur dann visualisiert (gerendert),
 wenn es sichtbare Elemente besitzt.

Ein HTML-Element beim IE und NS:

als HTML-Tag: Der IE und NS unterstützen z.T. verschiedene Tags
 als Objekt: Es gibt Objekte, die im IE **und** NS implementiert sind, aber z.T. auf verschiedene Weise (inkl. Syntax)

Collectionen zum Objekt document:

Zur Verwaltung des Dokumentes existieren Collectionen als Felder, die zugleich als Schnittstelle zum Programmierer dienen.
 Der Netscape besitzt teilweise Collectionen, die auch der Internet Explorer bedient.

Objekt document und Browserfenster:

Der **logische** Fenstername muss nur dann kodiert werden, wenn der Bezug nicht auf das aktuelle Dokument gehen soll.
 Der logische Fenstername stammt aus
 der Methode `.open()` zum `window` Objekt und kann ein frei festgelegter oder vordefinierter Name sein
 (vordefiniert bei automatischen Öffnen eines Fensters)
 aus den ID-Attributen bei FRAMESET mit FRAMES.

Die Kodierung des Zeigers von document kann dann entfallen, wenn das aktuelle Dokument referenziert wird.

Allerdings ist von der Kodierung ohne `window` bzw. ohne `window.document` abzuraten, denn



es könnten gleichnamige Eigenschaften und Methoden geben, die in anderen Objekten auch implementiert sind
es wird damit die Browserperformance gesenkt.

Zugriff:

window.document.eigenschaft	oder	document.eigenschaft	oder	eigenschaft
window.document.methode()	oder	document.methode()	oder	methode()

logischer_window_name.document.eigenschaft
logischer_window_name.document.methode()

logischer_window_name laut open()

Der **logische** Fenstername muss nur dann kodiert werden, wenn der Bezug nicht auf das Dokument im aktuellen Fenster gehen soll.

Der logische Fenstername stammt aus

der Methode .open() zum window Objekt und kann ein frei festgelegter oder vordefinierter Name sein
(vordefiniert bei automatischen Öffnen eines Fensters)
aus den ID-Attributen bei FRAMESET mit FRAMES.

Die Kodierung des Zeigers von document kann dann entfallen, wenn das aktuelle Dokument referenziert wird.

Allerdings ist von der Kodierung ohne window bzw. ohne window.document abzuraten, denn
es könnten gleichnamige Eigenschaften und Methoden geben, die in anderen Objekten auch implementiert sind
es wird damit die Browserperformance gesenkt.

Eigenschaften:

.alinkColor
.bgColor
.characterSet
.cookie
.defaultView
.dir
.doctype
.documentElement
.domain
.expando
.fgColor
.firstChild
.height
.implementation
.lastChild
.lastModified
.linkColor
.location
.localName
.namespaceURI
.nextSibling
.nodeName
.nodeType
.nodeValue
.ownerDocument
.parentNode
.prefix
.previousSibling
.referrer
.title
.URL
.vlinkColor
.width

Methoden:

.captureEvents()
.close()
.createElement()
.createTextNode()
.getElementById()
.getElementsByName()
.getElementsByTagName()
.mergeAttributes()
.open()
.releaseEvent()
.routeEvent()
.write()
.writeln()

4.3.2.1.3.1. Collectionen zum Objekt window.document des Netscape

Nachfolgende Collectionen dienen der Verwaltung des HTML-Dokumentes bzw. verschiedener HTML-Elemente.

In den Syntax-Beschreibungen wird aus Gründen der Vereinfachung immer vom aktuellen Fenster ausgegangen. Damit erfolgt nicht die Kodierung von `window.document` sondern nur `document`, da beide Formen für das aktuelle Fenster synonym sind.

4.3.2.1.3.1.1. **window.document.anchors** Collection des Netscape

Feld der Zeiger aller Anker im Dokument

Feldelementefolge laut HTML-Coding

Erzeugung:

durch den Browser

Beispiel für Anker:

```
<A      ID="ID_Anker"
      HREF="url"
      NAME="logischer_anker_name"
      TARGET="logischer_window_name"

>
      anker_text
</A>
```

Hinweis zu HREF= :

kann leer sein	per	Leerkette ""
	oder	"javascript:void(0);"

zu Anspringen eines Ankers:

den Wert laut NAME ablegen in	
<code>window.location.hash</code>	ohne vorgesetztes #
<code>window.location.href</code>	mit vorgesetztem #

Syntax:

```
[ var ZeigerAufFeld = ] document.anchors
[ var ZeigerAufFeldElement = ] document.anchors[index]
```

index	Integer ab 0
	muss in [] kodiert sein

`ZeigerAufFeldElement.eigenschaft`

`ZeigerAufFeldElement.methode()`

Eigenschaften:

`.length` Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.2. **window.document.applets** Collection des Netscape

Feld der Zeiger aller Applet-Objekte im Dokument

Elementefolge laut HTML-Coding

Applet muss alle Eigenschaften und Methoden, die im HTML-Dokument benutzt werden sollen, als **public** deklarieren

Erzeugung:

durch den Browser

Java-Applet (*.class) in HTML einbinden:

Beispiel:

```
<APPLET ID="ID_Applet"
      CODE="class_datei"
      CODEBASE="quellverzeichnis"
      NAME="Name_Applet"
      HEIGHT=applet_fenster_hoehe_in_pixel
      WIDTH=applet_fenster_breite_in_pixel
      MAYSCRIPT=true oder false
      ALT="alternativer_text"
      ALIGN=ausrichtung
      HSPACE=fenster_abstand_links_und_rechts_von_umgebung
      VSPACE=fenster_abstand_loben_und_unten_von_umgebung

>
      <PARAM Name="parameter_name"
              VALUE=parameter_wert
      >
      .....
</APPLET>
```

Syntax:

```
[ var ZeigerAufFeld = ] document.applets
[ var ZeigerAufFeldElement = ] document.applets[index]
```

index	Integer ab 0
	muss in [] kodiert sein



ZeigerAufFeldElement.eigenschaft
 ZeigerAufFeldElement.methode()

document.ID_Applet.eigenschaft
 document.ID_Applet.methode()

document.Name_Applet.eigenschaft
 document.Name_Applet.methode()

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.3. window.document.cookie Collection des Netscape (Cookie-Verwaltung im HTML-Dokument)

Feld der Zeiger auf alle Cookies zum Dokument

Ein Cookie kann bis zu 4 KBytes groß sein.

Es sind bis zu 300 verschiedene Cookies speicherbar.

Wenn Cookieverwaltung nicht erwünscht wird, so muss sie im Browser abgeschaltet werden --> eventuell laufen dann Webseiten nicht mehr korrekt ! Alternativ: Nach der Onlinesitzung die regelmäßige Löschung des **Inhaltes** des Cookie-Ordners **und** des Browser-Cache (Ordner selbst auf keinen Fall löschen !).

Nutzung Firewall

Cookies haben den gleichen Aufbau aber verschiedene Inhalte.

Erzeugung:

durch den Browser

Zugriff:

```
[ var ZeigerAufFeld = ] document.cookie
[ var ZeigerAufFeldElement = ] document.cookie[index]
```

index Integer ab 0
 muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

Methoden:

keine

Auf den Wert des Cookies sind alle Methoden des String-Objektes anwendbar, da der Wert immer eine Zeichenkette ist.

Aufbau eines Cookie:

Cookies haben den gleichen Aufbau aber verschiedene Inhalte.

Optionale Eigenschaften müssen ein Semikolon vorgesetzt bekommen und können beliebig kombiniert werden.

Mindestdeklaration eines Cookie (Wert eines Cookies):

```
document.cookie[0] = wert_des_cookie_1;
```

oder

```
var cookie_feld = document.cookie;
cookie_feld[0] = wert_des_cookie_1;
```

wert_des_cookies: Zeichenkette
 kann HTML-gerecht kodierte Sonderzeichen enthalten
 darf kein Blank enthalten, da jedes Cookie intern als Endekennzeichen ein Blank besitzt

Aufbau: freierwert[eigenschaften_liste]

Semikolon trennt die optionale Eigenschaftenliste ab
 Eigenschaften in der Liste durch Semikolon trennen

freierwert: cookie_titel=freier_wert

= ist festkodiert

Bsp: "Filmbeginn=23:15;path=...;expires=....."

Anwendungsbereich des Cookie also nicht die Lage auf der Festplatte der Users:

path=pfad_angabe

path= ist festkodiert

pfad_angabe:

Bsp.: 'Otto=der_erste;c:\privat\html_files\
 also Anwendbarkeit nur möglich, wenn HTML-Dokument
 innerhalb bzw. unterhalb von c:\privat\html_files liegt
 wenn nicht kodiert, wo wird der Pfad des Browser-Cache verwendet



Servergruppzugehörigkeit:**domain**=server_name_fragment**domain**= ist festkodiert

server_name_fragment:

Bsp.: "Otto=der_erste;domain=online.de"
 also alle Server, die im Namen den Rest
 online.de haben

wenn nicht kodiert, so wird der volle Name
 von dem Server abgelegt, der das Cookie
 verwaltet und nutzt

Verfallsdatum:**expires**=datum_angabe**expires**= ist festkodiert

datum_angabe: wochentag, tt-mm-jj hh-mm-ss
 nach Greenwich-Time

Bsp.: var loeschdatum = new Date();
 "Otto=der_erste;epires= " +
 loeschdatum.toGMTString;

wenn nicht kodiert, so Cookie mit Ende der
 Onlinesitzung von der Festplatte des
 Users automatisch gelöscht
 wenn kodiert, so am Verfallsdatum bzw. danach
 von der Festplatte des Users automatisch
 gelöscht (danach, wenn der User den PC
 nach dem Verfallsdatum erst wieder
 nutzt)

Cookie nur senden, wenn HTTPS-Protokoll also sichere Verbindung benutzt wird:**secure**

kein Wert

Beispiel für Cookie verwalten:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookien_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;
```



```

// CookieFeld auslesen und auf Cookie laut Name prüfen
var Gefunden=false;
var Index = 0;
do
{
    // aktuelles Cookie lesen
    CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

    // und auf Name prüfen
    if (Name == CookieNameUndWertAlsFeld [0])
    {
        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

return CookieWert;
}
</SCRIPT>

```

4.3.2.1.3.1.4. **window.document.embeds Collection des Netscape**

Feld der Zeiger aller als Plugin per EMBED-Tag eingebetteten Objekte im Dokument (inkl. Applets)
Elementefolge laut HTML-Coding

Diese Collection ist ein Alias für die plugins Collection **nur** bezüglich von Plugins (und nicht anderen einbettbaren Objekten), wobei letztere nur der Kompatibilität mit anderen plugin-fähigen Browsern dient. Alle eingebetteten Objekte haben in document.embeds ihren Zeiger (inklusive Plugins, Ausnahme: siehe tiefer).

Das Ansprechen von Plugins kann über die Collection navigator.mimeTypes per Eigenschaft .enabledPlugin erfolgen, die einen Zeiger auf das installierte Plugin enthält (wenn nicht installiert, so null-Zeiger). Wenn der Zeiger nicht null ist, dann sind die plugin-eigenen Methoden und Eigenschaften über Punktnotation ansprechbar.

Erzeugung:

durch Browser

Beispiel:

```

<EMBED
    SRC="url"
    TYPE="mime_typ"           // z.B. "image/gif"
    PLUGINSOURCE="plugin_url"
    HEIGHT="hoehe_plugin_objekt"
    WIDTH="breite_plugin_objekt"
    NAME="ID_Embed"
    HIDDEN="true oder false"  // true so Objekt unsichtbar
>
<PARAM Name="parameter_name" VALUE=parameter_wert>
.....
</EMBED>

```

Syntax:

```

[ var ZeigerAufFeld = ] document.embeds
[ var ZeigerAufFeldElement = ] document.embeds[index]

```

index Integer ab 0
 muss in [] kodiert sein

ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.5. **window.document.forms Collection des Netscape**

Feld der Zeiger aller Formular-Objekte im Dokument
Elementefolge laut HTML-Coding

Syntax:

```
document..forms[index]
```

Index Integer, ab 0
 muss in [] kodiert sein

Eigenschaften:



.length Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.6. window.document.frames Collection des Netscape

Feld der Zeiger aller Frames

Elementfolge laut HTML-Coding

Element ist eine Referenz auf das Fenster mit Frame-Eigenschaften

Syntax:

```
[ var ZeigerAufFeld = ] document.frames
[ var ZeigerAufFeldElement = ] document.frames[index]
```

index Integer ab 0
muss in [] kodiert sein

Beispiel für Inhalte zweier Frames tauschen:

Kodierung im Dokument, dass die beiden Frames definiert !

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
                          logischer_name_rahmen2,html_datei2
                          )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>
```

```
<A HREF="javascript:parent.rahmentausch(0, 'a.html', 1,b.html)">tauschen</A>
```

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.7. window.document.images Collection des Netscape

Feld der Zeiger aller img Objekte (auch INPUT mit Image)

Elementfolge laut HTML-Koding

Syntax:

```
document.images[index]
```

index Integer, ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

4.3.2.1.3.1.8. window.document.layers Collection des Netscape (nicht mehr ab Version 6.x)

Feld der Zeiger aller Layer-Objekte im Dokument

Syntax:

```
document.layers[index]
```

index Integer, ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente, ab 1

Beispiele:

unverschachtelter Layer:

```
<LAYER ID="ID_Layer" STYLE="position: absolute;">Ich bin eine Ebene</LAYER>
```

Layer-Zugriff per Script:

```
document. ID_Layer
oder document.layers["ID_Layer"]
oder document.layers[0]
```

Verwendung eines numerischen Indexes:

Die Indexfolge ist nicht die Folge laut HTML-Kodierung (außer Indexwert 0 = unterster Layer)
dafür z-index-Attribut kodieren und dieses verwenden

Anzahl der obersten Ebenen auf gleichem Level also ohne deren innere Layer:

```
document.layers.length
```

verschachtelte Layer:

```
<LAYER ID="ebene1" STYLE="position: absolute; left: 100; top: 100;">
```



```

    Nur ein Test...
    <LAYER ID="ebene2" STYLE="position: absolute; left: 50; top: 50;">
        Jaja!
    </LAYER>
</LAYER>

```

es wird vererbt: Bsp.: Verschiebung des äußeren Layers (Eltern) verschiebt den inneren (Kind),
wobei innerer in gleicher relativer Position zum äußeren bleibt wie vor der Verschiebung.

document.layers.length liefert 1

Zugriff auf inneren Layer:
document.ebene1.document.ebene2

Den Inhalt von Ebenen ändern:

```

document.ebene1.document.open();           // anzeigen
document.ebene1.document.write("Ein neuer Inhalt"); // füllen
document.ebene1.document.close();          // und schliessen
                                           // bleibt aber sichtbar

```

```

oder
with(document.ebene1.document.ebene2)
{
    open();
    write("So geht's auch");
    close();
}

```

Layer dynamisch erzeugen NACH dem kompletten Laden des Dokumentes
z.B. per <BODY onload=>

Achtung:

Ein Versuch, Ebenen direkt im HEAD per Script zu erzeugen, wird einfach ignoriert, wenn der BODY-Teil des HTML-Dokumentes nicht bereits komplett geparkt wurde.

Bsp.:

```

var neueEbene=new Layer(breite_in_Pixel);           // instanzieren

neueEbene.document.open();                         // anzeigen
neueEbene.document.write("test");                  // und füllen
neueEbene.document.close();                        // und schliessen
                                                    // bleibt aber sichtbar !!

```

Inhalt eines Layers:

```

laden      Aufruf der Methode "load" oder neueEbene.src="seite1.html";
anzeigen   neueEbene.visibility="show";

```

verschachtelte Layer dynamisch erzeugen

```

var neueEbene=new Layer(breite_in_Pixel);           // instanzieren
var neueEbene1=new Layer(breite_in_Pixel, document.neueEbene);

```

Eigenschaften von Layer lesen und/oder ändern

```

z.B.      X-Position      document.neueEbene.left=200; // Pixel
          Y-Position      document.neueEbene.top= 300; // Pixel

```

Layer verschieben:

```

Bsp.      document.ebene1.moveBy(50, 20);
          oder document.ebene1.left+=50; document.ebene1.top+=20;

```

4.3.2.1.3.1.9. window.document.links Collection des Netscape

Feld der Zeiger aller Objekte mit HREF-Eigenschaft (Attribut) sowie aller AREA-Objekte im Dokument
Elementefolge laut HTML-Coding

Syntax:

```
document.links[index]
```

```

index      Integer, ab 0
           muss in [ ] kodiert sein

```

Eigenschaften:

```
.length      Anzahl der Feldelemente, ab 1
```

4.3.2.1.3.1.10. window.document.plugins Collection des Netscape

siehe navigator.plugins Collection des Netscape



4.3.2.1.3.2 Objekte des HTML-Dokumentes des Netscape (Auswahl)

In den Beschreibungen wird aus Gründen der Vereinfachung immer vom aktuellen Fenster ausgegangen. Damit erfolgt nicht die Kodierung von `window.document` sondern nur `document`, da beide Formen für das aktuelle Fenster synonym sind.

Objekte sind in Script z.B. per `document.write()` bzw. `document.writeln()` oder per HTML-DOM-Methoden erzeugbar.

4.3.2.1.3.2.1. `window.document.body` Objekt des Netscape

Rumpf des HTML-Dokumentes

Erzeugung in HTML:

Beispiel:

```
<BODY ....>
      ....
</BODY>
```

Zugriff:

```
document.body.eigenschaft
document.body.methode()
document.ID_Body.eigenschaft
document.ID_Body.methode()
```

Hinweise zu den Methoden `document.write()` und `document.writeln()`:

Diese Methoden schreiben immer in den komplett geparsten BODY-Teil des Dokumentes und können neben Text und HTML-Code auch Scriptcode schreiben. Im Falle von Scriptcode gilt: Mit der Abarbeitung dieser Methoden bewirkt das Schreiben des Scriptcodes in den Body immer die **sofortige** Ausführung des Scriptcodes (analog zur Methode `eval()`, die aber keinen HTML-Code ausführen kann).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die **zuletzt** während der Laufzeit getätigte Wertsetzung zu einem Attribut definiert dessen **aktuellen** Attributwert.

4.3.2.1.3.2.2. `window.document.frame` Objekt des Netscape

Erzeugung unter HTML:

Beispiel:

```
<FRAMESET
    ROWS="zeilen_liste" oder COLS="spalten_liste" // Zeilenliste=Liste der Framehöhen
                                                    // mit Kommatrennung
                                                    // Spaltenliste=Liste der Framebreiten
                                                    // mit Kommatrennung
    onLoad="evenhandler1" // Anweisungen aktiviert NACH laden ALLER Frames
    onUnload="eventhandler2"
    onerror="eventhandler3"
    onBlur="eventhandler4"
    onFocus="eventhandler5"
>
[<FRAME
    SRC="frame_url" // url der HTML-Seite als Frameinhalt
    NAME="logischer_frame_name"
>]
.....
</FRAME>
.....
</FRAMESET>
```

Rahmen zwischen den einzelnen Fenstern des Framesets entfernen:

```
<FRAMESET    BORDER="0"
              FRAMEBORDER="0"
              FRAMESPACING="0"
              .....
>
```

Hinweise:

BORDER	bei Netscape	
	Breite des Rahmens	
	>= 0 Pixel	
FRAMEBORDER	bei IE	
	Rahmenanzeige	
	0 oder "no"	aus
	1 oder "yes"	ein
FRAMESPACING	bei IE	



Rahmenbreite
 >=0 Pixel

Zugriff:

document.ID_Frame.eigenschaft
 document.ID_Frame.methode()

document.frames[index].eigenschaft
 document.frames[index].methode()

index: ab 0
 muss in [] kodiert sein

Eigenschaften (Auswahl):

.name entspricht NAME
 .length entspricht logischer_window_name.frames.length
 .parent Fenster, das das Frame-Objekt erhält
 .self aktueller Frame

Methoden (Auswahl):

.blur() deaktiviert Frame
 .focus() aktiviert Frame

Eventhandler (Auswahl):

onLoad
 onUnload
 onMove
 onResize

Collectionen:**window.document.frames Collection**

Feld der Zeiger aller Frames
 Elementefolge laut HTML-Coding
 Element ist eine Referenz auf das Fenster mit Frame-Eigenschaften

Syntax:

```
[ var ZeigerAufFeld = ] document.frames
[ var ZeigerAufFeldElement = ] document.frames[index]
```

index Integer ab 0
 muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Möglichkeiten des Laden eines Dokumentes:

Laden eines fremden Dokumentes innerhalb eines Frame ist rechtlich nicht zulässig (Abmahnungsgefahr), wenn der Eigentümer der Fremdseite nicht explizit zugestimmt hat.

Laden eines fremden Dokumentes ohne Framedarstellung:

```
<HTML>
  <HEAD>
    <SCRIPT LANGUAGE="JavaScript">
      <!--
        function laden()
        { location.href = "http://www.test.de";}
      // -->
    </SCRIPT>
  </HEAD>

  <BODY>
    <FORM>
      <INPUT TYPE="button"
        VALUE="www.test.de anwaehlen "
        onclick="laden()">
    </FORM>
  </BODY>
</HTML>
```

Eigenes Dokument wird durch fremde Webseite geladen:***CopyRight-Meldung auf fremden Host erzeugen:***

Beispiel 1:

```
// In diesem Beispiel schreibt das Script Text auf die "entführte" Seite!
var HostUrl="www.test.de";
```



```

if (    (parent !=null)
    &&  (parent != self)
)
{
    var host=parent.location.hostname;

    if(host != HostUrl)
    {
        document.write(
            "Diese Seite wurde ausgeliehen bei "
            + "<A HREF=" + location.href
            + " TARGET='_parent'"
            + ">"
            + HostUrl
            + "</A>"
        );
    }
}

```

Beispiel 2:

```

<BODY>
.....
if (    (parent != null)
    &&  (parent != self)
)
{
    var  meine_url = "www.test.de"
    var  mein_host_name = "http://" + meine_url;

    var  fremder_host_name = parent.location.hostname;

    if ( fremder_host_name != mein_host_name)
    {
        document.write(      " Diese Seite liegt auf "
                               + "<A HREF=" + location.href + "\"\"
                               + " TARGET=\"_parent\"\"
                               + ">"
                               + "</A>"
                               + " und stammt von "
                               + mein_host_name
        );
    }
}
.....
</BODY>

```

Framedarstellung der eigenen Webseite sofort und ohne Bildschirmmeldung aktivieren:

Dieses Coding muss in jedes HTML-Dokument, das in einem Frame geladen wird. Bei Einzelaufwurf des Dokumentes wird automatisch die Seite mit dem FRAMSET aktiviert, also die vollständige Framedarstellung.

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
    var EigenerHost="http://www.test.de";
        // window.location.hostname ist Leerkette, wenn Browser offline
    var StartSeite="_start.html"; // muss das FRAMESET enthalten !

    function InaktiveFrameDarstellungAktivieren()
    {
        if (top.frames.length == 0)
        {
            // keine Frame-Darstellung aktiv, also diese aktivieren
            top.location.href = StartSeite;
        }
    }

    if (    (parent != null) // Elternobjekt existiert
        && (parent != self) // Eltern sind vorhanden Eltern: dieses Dokument (self) ist ein Kind
    )

```



```

    )
    {
        // aktuellen ElternHost ermitteln
        var ElternHost=parent.location.hostname;

        // ElternHost prüfen ob eigener und nicht leer, also Browser online ist
        if ( (ElternHost != "") // Browser ist online
            && (ElternHost != EigenerHost)
        )
        {
            // fremder Host, also als oberstes Fenster nun die Startseite anzeigen
            // und damit den eigenen Host aktivieren
            top.location.href=EigenerHost + '/' + Startseite;
        }
        else
        {
            // eigener Host und/oder Browser ist offline
            InaktiveFrameDarstellungAktivieren();
        }
    }
    else
    {
        // Elternobjekt existiert nicht und/oder dieses Dokument (self) ist kein Kind
        InaktiveFrameDarstellungAktivieren();
    }
}
//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Reload eines Dokumentes mit allen seinen FRAMES:

```

function frame_neu_laden()
{
    for (var i=0; i<windows.FRAMES.length; i++)
    { windows.frames[i].location.reload(false); }
}

<FRAMESET onResize="frame_neu_laden()">

```

Datei ohne FRAMESET in eine Datei mit FRAMESET laden

```

if (self.location == top.location)
{ location.href="/FRAMESET.html?" + escape(location.pathname); }

```

Mehrere Frameinhalte gleichzeitig ändern:**Inhalte zweier Frames tauschen:**

Kodierung im Dokument, dass die beiden Frames definiert !

```

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
                          logischer_name_rahmen2,html_datei2
                          )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch(0, 'a.html', 1,b.html)">tauschen</A>

```

Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framenamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument. Argumententrenner sind Doppelpunkte.



Tausch:

erster Frame retten und dann mit zweiten Frame überschreiben
 zweiten Frame mit drittem Frame überschreiben

 vorletzten Frame mit letztem Frame überschreiben
 letzten Frame mit geretteten ersten Frame überschreiben

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)           // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)      // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);

                for(var i = 0; i < arguments.length; i++)    // Index ab 0
                {
                    pos_doppelpunkt = arguments[i].indexOf(":");

                    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
                    {
                        // ersten zu ersetzenden Framenamen retten

                        if (i=0)
                        { var rette_logischer_framename =
                          arguments[i].substring(0, pos_doppelpunkt)
                        }

                        // tauschen der logischen Framenamen
                        logischer_framename_zu_ersetzer_frame =
                          arguments[i].substring(0, pos_doppelpunkt);

                        logischer_framename_ersetzer_frame =
                          arguments[i].substring(0, pos_doppelpunkt + 1);

                        frames[logischer_framename_zu_ersetzer_frame].location.href =
                          logischer_framename_ersetzer_frame;
                    }
                }

                frames[logischer_framename_ersetzer_frame].location.href=
                    rette_logischer_framename;
            }
        }
    }

// -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>
```

Frame und Datenaustausch:

Zwischen Frames können Daten ausgetauscht werden.

Beispiel Adressierung eines Frame über das FRAMESET

```
<FRAMESET .... >
  <FRAMESET ..... >
    <FRAME SRC="test.html" NAME="test">
    <FRAME SRC="test1.html" NAME="test1">
  </FRAMESET>
  <FRAME SRC="test2.html" NAME="test2">
</FRAMESET>
```

```
parent.test2.location.href="neu.html";    // Laden von neu.html in den Frame test2
```

Beispiel: Auf Objekte im FRAMESET-Dokument durch ein FRAME-Dokument zugreifen



```
im FRAMESET-Dokument wird kodiert
```

im FRAME-Dokument wird auf Kette zugegriffen: `alert(parent.Kette);`

Beispiel: Auf Objekte im FRAME-Dokument durch das FRAMESET-Dokument zugreifen

im FRAME-Dokument wird kodiert	var Kette="Hallo !"; <FRAME ...NAME ="test2" ...>
--------------------------------	--

im FRAMESET-Dokument wird auf Kette zugegriffen: `alert(parent.test2.Kette);`

Beispiel: Textdaten-Übergabe durch an Url angehängte Textdaten

```

quelle.htm:      lädt ziel.htm
                   übergibt Textdaten an ziel.htm

```

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ziel_seite_laden_mit_datenuebergabe(uebergabe_datan)
{location.href = "ziel.htm?" + escape(uebergabe_datan);}
// Url von ziel.htm als Suchbegriff mit angehangenen Daten merken, die per escape() in das
// Url-Format konvertiert wurden
// -->
</SCRIPT>
</HEAD>

<BODY>
An ziel.htm zu &uuml;bergebenden Text eingeben:
<FORM>
    <TEXTAREA      NAME=eingabe
                  ROWS=5
                  COLS=40
    >
    </TEXTAREA>
    <BR>
    <INPUT  TYPE=button
            VALUE="Zielseite laden"
            onClick=" ziel_seite_laden_mit_datenuebergabe(this.form.eingabe.value)">

</FORM>
</BODY>
</HTML>
```

ziel.htm: wird durch quelle.htm geladen
erhält Textdaten von quelle.htm

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    datenuebernahme()
    {
        // Url mit angehangenen Daten holen
        // search liefert ?daten
        uebernahme_datens= location.search;

        // ? abschneiden
        uebernahme_datens= uebernahme_datens.substring(1, uebernahme_datens.length);

        // unescape: von Url-Format nach Zeichenkette
        document.ausgabe.ausgabefeld.value=unescape(uebernahme_datens);
    }

// -->
</SCRIPT>
</HEAD>

<BODY onLoad=" datenuebernahme ()">
Von quelle.htm &uuml;bernommener Text:
```



```

        <FORM NAME=ausgabe>
            <TEXTAREA NAME=ausgabefeld
                ROWS=10 COLS=40>
            </TEXTAREA>
        </FORM>
    </BODY>
</HTML>

```

Beispiel: Textdaten-Übergabe durch Fenster-Handle

Die Textdaten und die der Javascript-Code, der die Textdaten verarbeitet, sind **getrennte** HTML-Dokumente. Nachteil dieser Variante ist, dass in beiden Dokumenten die logischen Namen vom Formular und dem Textarea identisch kodiert werden müssen, also eine doppelte Verwaltung nötig ist.

HTML-Dokument, das die Textdaten enthält:

```

<HTML>
<BODY>
    <FORM NAME=formular>
        <TEXTAREA NAME=text_area>
            // Hier die Textdaten als Wert von TEXTAREA
        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Dokument, das die Textdaten verarbeitet:

```

<HTML>
<HEAD>
    <SCRIPT LANGUAGE="JavaScript">
<!--
    function TextAreaWertLesen(url)
    {
        // url enthält den Pfad und den Namen des Dokumentes, dass die
        // Textarea-Werte enthält
        var handle = window.open(url_der_datendatei,"", "width=100,height=100");

        // Handle erzeugen und Fenster mit dem Textarea anzeigen
        // (Fenstergröße ist egal)
        var data = handle.document.forms[formular].elements[text_area].value;

        handle.close();

        return data;
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

4.3.2.1.3.2.3. window.document.frameset Objekt des Netscape

Erzeugung unter HTML:

Beispiel:

```

<FRAMESET
    ROWS="zeilen_liste" oder COLS="spalten_liste" // Zeilenliste=Liste der Framehöhen
                                                    // mit Kommatrennung
                                                    // Spaltenliste=Liste der Framebreiten
                                                    // mit Kommatrennung
                                                    // Anweisungen aktiviert NACH laden ALLER Frames

    onLoad="evenhandler1"
    onUnload="eventhandler2"
    onerror="eventhandler3"
    onBlur="eventhandler4"
    onFocus="eventhandler5"

>

[<FRAME
    SRC="frame_url" // url der HTML-Seite als Frameinhalt
    NAME="logischer_frame_name"

>]
.....

```



```

</FRAME>
.....
</FRAMESET>

```

Rahmen zwischen den einzelnen Fenstern des Framesets entfernen:

```

<FRAMESET      BORDER="0"
                FRAMEBORDER="0"
                FRAMESPACING="0"
                .....
>

```

Hinweise:

BORDER	bei Netscape Breite des Rahmens >= 0 Pixel	
FRAMEBORDER	bei IE Rahmenanzeige 0 oder "no" aus 1 oder "yes" ein	
FRAMESPACING	bei IE Rahmenbreite >=0 Pixel	

Eigenschaften (Auswahl):

.name	entspricht NAME
.length	entspricht logischer_window_name.frames.length
.parent	Fenster, das das Frame-Objekt erhält
.self	aktueller Frame

Methoden (Auswahl):

.blur()	deaktiviert Frame
.focus()	aktiviert Frame

Eventhandler (Auswahl):

onLoad
onUnload
onMove
onResize

Collectionen:

window.document.frames

Feld der Zeiger aller Frames

Elementefolge laut HTML-Coding

Element ist eine Referenz auf das Fenster mit Frame-Eigenschaften

Syntax:

```

[ var ZeigerAufFeld = ] document.frames
[ var ZeigerAufFeldElement = ] document.frames[Index]

```

Index	Integer ab 0 muss in [] kodiert sein
-------	--

Eigenschaften:

.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
---------	--

Möglichkeiten des Laden eines Dokumentes:

Laden eines fremden Dokumentes innerhalb eines Frame ist rechtlich nicht zulässig (Abmahnungsgefahr), wenn der Eigentümer der Fremdseite nicht explizit zugestimmt hat.

Laden eines fremden Dokumentes ohne Framedarstellung:

```

<HTML>
  <HEAD>
    <SCRIPT LANGUAGE="JavaScript">
      <!--
        function laden()
        { location.href = "http://www.test.de"; }
      // -->
    </SCRIPT>
  </HEAD>

  <BODY>
    <FORM>
      <INPUT TYPE="button"
        VALUE="www.test.de anwählen "

```



```

        onclick="laden()">
    </FORM>
</BODY>
</HTML>

```

Eigenes Dokument wird durch fremde Webseite geladen::

CopyRight-Meldung auf fremden Host erzeugen:

Beispiel 1:

```

// In diesem Beispiel schreibt das Script Text auf die "entführte" Seite!
var HostUrl="www.test.de";

if (    (parent !=null)
    &&  (parent != self)
    )
{
    var host=parent.location.hostname;

    if(host != HostUrl)
    {
        document.write(
            "Diese Seite wurde ausgeliehen bei "
            + "<A HREF=" + location.href
            + " TARGET='_parent'"
            + ">"
            + HostUrl
            + "</A>"
        );
    }
}

```

Beispiel 2:

```

<BODY>
.....
if (    (parent != null)
    &&  (parent != self)
    )
{
    var  meine_url = "www.test.de"
    var  mein_host_name = "http://" + meine_url;

    var  fremder_host_name = parent.location.hostname;

    if ( fremder_host_name != mein_host_name)
    {
        document.write(
            " Diese Seite liegt auf "
            + "<A HREF=" + location.href + "\"
            + " TARGET='_parent'"
            + ">"
            + "</A>"
            + " und stammt von "
            + mein_host_name
        );
    }
}
.....
</BODY>

```

Framedarstellung der eigenen Webseite sofort und ohne Bildschirmmeldung aktivieren:

Dieses Coding muss in jedes HTML-Dokument, das in einem Frame geladen wird. Bei Einzelaufwurf des Dokumentes wird automatisch die Seite mit dem FRAMSET aktiviert, also die vollständige Framedarstellung.

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
    var EigenerHost="http://www.test.de";

```




```

        // window.location.hostname ist Leerkette, wenn Browser offline
var StartSeite="_start.html";

function InaktiveFrameDarstellungAktivieren()
{
    if (top.frames.length == 0)
    {
        // keine Frame-Darstellung aktiv, also diese aktivieren
        top.location.href = StartSeite; // muss das FRAMESET enthalten !
    }
}

if ( (parent != null) // Elternobjekt existiert
    && (parent != self) // Eltern sind vorhanden Eltern: dieses Dokument (self) ist ein Kind
    )
{
    // aktuellen ElternHost ermitteln
    var ElternHost=parent.location.hostname;

    // ElternHost prüfen ob eigener und nicht leer, also Browser online ist
    if ( (ElternHost != "") // Browser ist online
        && (ElternHost != EigenerHost)
        )
    {
        // fremder Host, also als oberstes Fenster nun die Startseite anzeigen
        // und damit den eigenen Host aktivieren
        top.location.href=EigenerHost + '/' + StartSeite;
    }
    else
    {
        // eigener Host und/oder Browser ist offline
        InaktiveFrameDarstellungAktivieren();
    }
}
else
{
    // Elternobjekt existiert nicht und/oder dieses Dokument (self) ist kein Kind
    InaktiveFrameDarstellungAktivieren();
}
}

//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Reload eines Dokumentes mit allen seinen FRAMES:

```

function frame_neu_laden()
{
    for (var i=0; i<windows.FRAMES.length; i++)
    { windows.frames[i].location.reload(false); }
}

<FRAMESET onResize="frame_neu_laden()">

```

Datei ohne FRAMESET in eine Datei mit FRAMESET laden

```

if (self.location == top.location)
{ location.href="/FRAMESET.html?" + escape(location.pathname); }

```

Mehrere Frameinhalte gleichzeitig ändern:**Inhalte zweier Frames tauschen:**

Kodierung im Dokument, dass die beiden Frames definiert !

```

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
        logischer_name_rahmen2,html_datei2

```



```

    )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch(0, 'a.html', 1,b.html)">tauschen</A>

```

Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framenamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument. Argumententrenner sind Doppelpunkte.

Tausch:

```

    erster Frame retten und dann mit zweiten Frame überschreiben
    zweiten Frame mit drittem Frame überschreiben
    .....
    vorletzten Frame mit letztem Frame überschreiben
    letzten Frame mit geretteten ersten Frame überschreiben

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)           // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)     // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);

                for(var i = 0; i < arguments.length; i++)    // Index ab 0
                {
                    pos_doppelpunkt = arguments[i].indexOf(":");

                    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
                    {
                        // ersten zu ersetzenden Framenamen retten

                        if (i==0)
                        { var rette_logischer_framename =
                            arguments[i].substring(0, pos_doppelpunkt)
                        }

                        // tauschen der logischen Framenamen
                        logischer_framename_zu_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt);

                        logischer_framename_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt + 1);

                        frames[logischer_framename_zu_ersetzender_frame].location.href =
                            logischer_framename_ersetzender_frame;
                    }
                }

                frames[logischer_framename_ersetzender_frame].location.href=
                    rette_logischer_framename;
            }
        }
    }
// -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>

```

Frame und Datenaustausch:

Zwischen Frames können Daten ausgetauscht werden.



Beispiel Adressierung eines Frame über das FRAMESET

```
<FRAMESET .... >
  <FRAMESET ..... >
    <FRAME SRC="test.html" NAME="test">
    <FRAME SRC="test1.html" NAME="test1">
  </FRAMESET>
  <FRAME SRC="test2.html" NAME="test2">
</FRAMESET>
```

```
parent.test2.location.href="neu.html";    // Laden von neu.html in den Frame test2
```

Beispiel: Auf Objekte im FRAMESET-Dokument durch ein FRAME-Dokument zugreifen

```
im FRAMESET-Dokument wird kodiert      var Kette="Hallo !";

im FRAME-Dokument wird auf Kette zugegriffen:  alert(parent.Kette);
```

Beispiel: Auf Objekte im FRAME-Dokument durch das FRAMESET-Dokument zugreifen

```
im FRAME-Dokument wird kodiert      var Kette="Hallo !";
                                     <FRAME ...NAME ="test2" ...>

im FRAMESET-Dokument wird auf Kette zugegriffen:  alert(parent.test2.Kette);
```

Beispiel: Textdaten-Übergabe durch an Url angehängte Textdaten

**quelle.htm: lädt ziel.htm
 übergibt Textdaten an ziel.htm**

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function ziel_seite_laden_mit_datenuebergabe(uebergabe_daten)
    {location.href = "ziel.htm?" + escape(uebergabe_daten);}
      // Url von ziel.htm als Suchbegriff mit angehangenen Daten merken, die per escape() in das
      //          Url-Format konvertiert wurden

// -->
</SCRIPT>
</HEAD>

<BODY>
An ziel.htm zu &uuml;bergabenden Text eingeben:
<FORM>
  <TEXTAREA      NAME=eingabe
                  ROWS=5
                  COLS=40
  >
  </TEXTAREA>
  <BR>
  <INPUT  TYPE=button
          VALUE="Zielseite laden"
          onClick=" ziel_seite_laden_mit_datenuebergabe(this.form.eingabe.value)">

</FORM>
</BODY>
</HTML>
```

**ziel.htm: wird durch quelle.htm geladen
 erhält Textdaten von quelle.htm**

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    datenuebernahme()
    {
      // Url mit angehangenen Daten holen
      // search liefert ?daten
      uebernahme_daten= location.search;
```



```

        // ? abschneiden
        uebernahme_datens= uebernahme_datens.substring(1, uebernahme_datens.length);

        // unescape: von Url-Format nach Zeichenkette
        document.ausgabe.ausgabefeld.value=unescape(uebernahme_datens);
    }

// -->
</SCRIPT>
</HEAD>

<BODY onLoad=" datenuebernahme ()">
    Von quelle.htm &uuml;bernommener Text:
        <FORM NAME=ausgabe>
            <TEXTAREA      NAME=ausgabefeld
                                ROWS=10 COLS=40>
            </TEXTAREA>
        </FORM>
</BODY>
</HTML>

```

Beispiel: Textdaten-Übergabe durch Fenster-Handle

Die Textdaten und die der Javascript-Code, der die Textdaten verarbeitet, sind **getrennte** HTML-Dokumente. Nachteil dieser Variante ist, dass in beiden Dokumenten die logischen Namen vom Formular und dem Textarea identisch kodiert werden müssen, also eine doppelte Verwaltung nötig ist.

HTML-Dokument, das die Textdaten enthält:

```

<HTML>
<BODY>
    <FORM NAME=formular>
        <TEXTAREA NAME=text_area>
            // Hier die Textdaten als Wert von TEXTAREA
        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Dokument, das die Textdaten verarbeitet:

```

<HTML>
<HEAD>
    <SCRIPT LANGUAGE="JavaScript">
<!--
    function TextAreaWertLesen(url)
    {
        // url enthält den Pfad und den Namen des Dokumentes, dass die
        //      Textarea-Werte enthält
        var handle = window.open(url_der_datendatei,"", "width=100,height=100");

        // Handle erzeugen und Fenster mit dem Textarea anzeigen
        //      (Fenstergröße ist egal)
        var data = handle.document.forms[formular].elements[text_area].value;

        handle.close();

        return data;
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

4.3.2.1.3.2.4. window.document.img Objekt des Netscape

Erzeugung unter HTML:

Beispiele:

Bild nicht im Formular:

```
<IMG
```



```

SRC="url_des_bildes_in_voller_auflösung"
NAME="logischer_bild_name"
LOWSRC="url_des_bildes_in_geringer_auflösung"
ALT="alternativer_text"
ALIGN=ausrichtung
HEIGHT=hoehe
WIDTH=breite
BORDER=rahmenbreite
HSPACE=abstand_links_und_rechts_zur_umgebung
VSPACE=abstand_oben_und_unten_zur_umgebung
ISMAP
USEMAP=          "map_url#image_map"
           oder   "map_name"
onAbort="eventhandler1"
onerror="eventhandler2"
onLoad="eventhandler3"

```

>

Bild im Formular:

```

<INPUT TYPE=image
      SRC=url_des_bildes_in_voller_auflösung"...

```

>

Beispiel für Linie mit variabler Dimension:

Es wird ein Bild aus 1x1 neu dimensioniert, das eine kleine Dateigröße hat und nur 1 mal geladen werden muss.
Durch die Angaben von Breite und Höhe kann eine vertikale oder horizontale Linie erzeugt werden. Die
Linienpositionierung erfolgt per STYLE-Attribut.

horizontale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="300" HEIGHT="10" BORDER="0" ALT="" STYLE=" ....">
```

vertikale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="10" HEIGHT="300" BORDER="0" ALT="" STYLE=" ....">
```

Erzeugung in Script:

```
var Bild = new Image([breite, hoehe]);
```

erzeugt Instanz und lädt zugleich das Bild

zeigt das Bild NICHT an

Verwendung z.B. bei animiertem Bild, wobei die Animation geladene Bilder voraussetzt

Zugriff:

Bild nicht im Formular:

document.ID_Img.eigenschaft

document.images[index].eigenschaft

index: ab 0

muss in [] kodiert sein

Bild im Formular:

document.ID_Img.eigenschaft

document.images[index].eigenschaft

document.ID_Formular.elements[index].eigenschaft

index: ab 0

muss in [] kodiert sein

Bild in Script: per Variable aus new

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```

var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url;           // ohne "" kodieren da Zeiger
document.write(
    '<IMG NAME="IMG_Bild"'
    + ' SRC="' + BildObjekt.src + '"'
    + ' HEIGHT=' + Bild_Hoehe
    + ' WIDTH=' + Bild_Breite
    + '>'
);

```

Eigenschaften (ausgewählte):

.border

entspricht BORDER



	nur lesen
.complete	wenn mit true belegt, so Bild komplett geladen wenn mit false belegt, so Bild noch nicht komplett geladen
	nur lesen
.height	entspricht HEIGHT
	nur lesen
.hspace	entspricht HSPACE
	nur lesen
.lowsrc	entspricht LOWSRC
.name	entspricht NAME
	nur lesen
.src	entspricht SRC
.vspace	entspricht VSPACE; Standard ist 0
	nur lesen
.width	entspricht WIDTH
	nur lesen

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```
var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url; // ohne "" kodieren da Zeiger
document.write(
    '<IMG NAME="IMG_Bild"'
    + ' SRC="' + BildObjekt.src + "' '
    + ' HEIGHT=' + Bild_Hoehe
    + ' WIDTH=' + Bild_Breite
    + '>'
);
```

Events bei sich überlagernden Objekten:

Beispiel: Sollte ein Image mit seiner Erzeugung ein anderes Image überlagern, so ist die Eventübergabe von und an das untere Image unterbrochen: Ereignisse onXXX kommen nicht mehr durch, solange das untere Bild nicht den **Fokus** erhält. Alternativ ist die Style-Eigenschaft z-index zu kodieren und dann der z-index auf den obersten zu setzten.

Collectionen:

window.document.images Collection des Netscape

Feld der Zeiger aller img Objekte

Elementefolge laut HTML-Koding

Syntax:

```
document.images[index]
```

index Integer, ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

4.3.2.1.3.2.5. window.document.layer / window.document.ilayer Objekt des Netscape unter 6.x

Objekt eines LAYER bzw. ILAYER (ILAYER steht für inline Layer)

Netscape hat nur bis unter Version 6.x das Layer-Objekt implementiert.

HTML 4 unterstützt kein LAYER- und ILAYER-Tag mehr.

Die Tags LAYER und ILAYER werden vom Internet Explorer nicht erkannt.

Das layer Objekt untergliedert das HTML-Dokument in Ebenen, wobei diese sich wie Folien überlappen oder ganz verdecken können.
kann ein HTML-Dokument laden
kann einen Ausschnitt als Sichtfenster erzeugen

Das Layer-Objekt ist an sich **völlig veraltet**:

Foliendarstellung: Die Eigenschaften bezüglich Position des Layers, bezüglich Ausschnitt im Layer und bezüglich Überlagerung von Layern sind z.B. auch anhand von DIV's in Verbindung mit Style realisierbar. Vorteil: DIV und Style kennt jeder moderne Browser, dagegen Layer nicht.

Laden eines HTML-Dokumentes: Die Anzeige eines Fensters ohne Fensterelemente (beim Internet Explorer auch durch zusätzliche Fensterarten) kann einen Layer ersetzen.

Erzeugung unter HTML:

Beispiel:

```
<LAYER
    NAME="logischer_layer_name"
    WIDTH=breite_in_pixel
```

fuer Javascript oder siehe Attribute ABOVE und BELOW
falls kein Ausschnitt definiert wird, so Breite des Layers
falls Ausschnitt per CLIP definiert wird, so Ausschnittbreite > Breite
möglich



HEIGHT=hoehe_in_pixel	falls kein Ausschnitt definiert wird, so Höhe des Layers falls Ausschnitt per CLIP definiert wird, so Ausschnittshöhe > Höhe möglich
CLIP="x1,y1,x2,y2" oder "x,y"	Ausschnitt im Layer bzw. auf Layer und seine Umgebung Ausschnitt kann eigenartigerweise größer als Layerdimension sein x1,y1 linke obere Ecke des Ausschnitt-Fenster; x ist Spalte x1 und y1 sind RELATIV zur linken oberen Layerrand-Ecke x2,y2 rechte untere Ecke des Ausschnitt-Fenster; x ist Spalte x2 und y2 sind RELATIV zur rechten unteren Layerrand-Ecke x Breite in Pixel, kann > WIDTH sein y Höhe in Pixel, kann > HEIGHT sein
SRC="url_oder_dateiname"	im Layer anzuzeigende HTML oder GIF oder JPG-Datei
BGCOLOR=#rrggbb	Hintergrund-Farbe
BACKGROUND="url_oder_datei"	Hintergrund-Grafik-Datei
VISIBILITY="show"	innerhalb Überlappung immer anzeigen
oder "hide"	innerhalb Überlappung nicht anzeigen
oder "inherit"	nur anzeigen wenn Elternlayer auch angezeigt ist
ABOVE="logischer_name_des_layers_der_diesen_überlagern_soll"	nicht zusammen mit BELOW und Z-INDEX kodieren
BELOW="logischer_name_des_layers_der_von_diesem_überlagert_werden_soll"	nicht zusammen mit ABOVE und Z-INDEX kodieren
Z-INDEX=position_aus_ueberlagerungsfolge	unverschachtelter Layer: 1=unterster ... n=oberster verschachtelter Layer wenn >0, so 1=unterster .. n=oberster wenn <0, so -1=oberster ..-n=unterster nicht zusammen mit ABOVE und BELOW kodieren
>	
html_elemente_des_layer	
</LAYER>	

Beispiel:

<ILAYER> ... </ILAYER> analog zu <LAYER> ... </LAYER>

Erzeugung in Script:

var LayerObjekt = new Layer([breite_in_pixel [, zeiger_auf_eltern_layer]]);

Beispiel: var element_aussen = new Layer(300); // <LAYER>
 element_aussen.src="url";
 element_aussen.visibility="show";
 element_aussen.document.open();
 element_aussen.document.write('HALLO');
 element_aussen.document.close();
 element_innen = new Layer(100, document. element_aussen); // <ILAYER>

Zugriff:

document. **logischer_layer_name**.eigenschaft
 document. **logischer_layer_name**.methode()

document.layers[index].eigenschaft
 document.layers[index].methode()

index: ab 0
 muss in [] kodiert sein

Variable laut new Layer()

Eigenschaften (ausgewählte):

.above liefert den Zeiger vom nächsten übergeordneten Layer laut dessen NAME-Attribut
 liefert **nicht** zIndex
 Achtung: Die oberste Ebene ist immer das Browserfenster.
 Vergleich der Zeiger per
 if (id_des_fensters_aus_open == document.layer[index].above)
 { }
 siehe .zIndex
 nur lesen

.background Url einer Grafikdatei für den Hintergrund
 Pfadangabe möglich
 automatisches Kacheln
 "null" für kein Hintergrundbild
 lesen und schreiben



.below	liefert den Zeiger vom nächsten tieferen Layer laut dessen NAME-Attribut liefert nicht zIndex wenn "null" so kein nächst tieferer Layer vorhanden siehe .zIndex nur lesen
.bgColor	Hintergrundfarbe des Layers "#xxxxxx" z.B. "#FF0000" vordefiniert z.B. "red" "null" so Hintergrund transparent, also darunterliegende Ebene(n) schimmert(n) durch lesen und schreiben
.clip	Diese Eigenschaft dient zur Verwaltung eines Ausschnittes im Layer entspricht CLIP-Attribut und ist nur über die Untereigenschaften verwendbar: CLIP="x1,y1,x2,y2" x1,y1 linke obere Ecke des Ausschnitt-Fenster; x ist Spalte x1 und y1 sind RELATIV zur linken oberen Layerrand-Ecke x2,y2 rechte untere Ecke des Ausschnitt-Fenster; x ist Spalte x2 und y2 sind RELATIV zur rechten unteren Layerrand-Ecke x1 entspricht .clip.left y1 entspricht .clip.top x2 entspricht .clip.right y2 entspricht .clip.bottom Hinweis: möglichst nicht kodieren und dafür durch .style.clip zu ersetzen
.clip.bottom	y2 aus CLIP="x1,y1,x2,y2" lesen und schreiben
.clip.height	Höhe des Ausschnittes in Pixel y aus CLIP="x,y" oder Differenz aus y2 minus y1 aus CLIP="x1,y1,x2,y2" .clip.height = .clip.bottom minus .clip.top lesen und schreiben
.clip.left	x1 aus CLIP="x1,y1,x2,y2" lesen und schreiben
.clip.right	x2 aus CLIP="x1,y1,x2,y2" lesen und schreiben
.clip.top	y1 aus CLIP="x1,y1,x2,y2" lesen und schreiben
.clip.width	Breite des Ausschnittes in Pixel x aus CLIP="x,y" oder Differenz aus x2 minus x1 aus CLIP="x1,y1,x2,y2" .clip.width = .clip.right minus .clip.left lesen und schreiben
.height	Höhe des Layers lesen und schreiben
.document	Zeiger auf das HTML-Dokument im Layer
.left	X-Koordinate linke obere Layerecke in Pixel, auch negativ String Ziffernfolge eines Integer und "absolute" Pixelangabe bezüglich Fenster "relative" Pixelangaben bezüglich zum den umgebenden HTML-Element Hinweis: möglichst nicht kodieren und dafür durch STYLE zu ersetzen: position:absolute;left=.... bzw. position:relative;left=.... lesen und schreiben
.name	enthält logischen Namen des Layers laut NAME-Attribut im Tag <LAYER> bzw. <ILAYER> nur lesen
.pageX	Abstand in Pixel immer zum linken Rand des Fensters Hinweis: möglichst nicht kodieren und dafür durch STYLE zu ersetzen: position:absolute;left=.... lesen und schreiben
.pageY	Abstand in Pixel immer zum oberen Rand des Fensters



Hinweis: möglichst nicht kodieren und dafür durch STYLE zu ersetzen:

position:absolute;top=....

lesen und schreiben

.parentLayer	liefert den Zeiger auf den Eltern-Layer, der diesen Layer enthält. liefert nicht zIndex Eltern-Layer kann mehrere innere Layer besitzen. Die oberste Ebene ist immer das Browserfenster. nur lesen
.siblingAbove	liefert den Zeiger vom nächsten übergeordneten Layer, der innerhalb eines gemeinsamen Eltern-Layers liegt liefert nicht zIndex ist "null" wenn kein übergeordneter Layer vorhanden ist nur lesen
.siblingBelow	liefert den Zeiger vom nächsten untergeordneten Layer, der innerhalb eines gemeinsamen Eltern-Layers liegt liefert nicht zIndex ist "null" wenn kein untergeordneter Layer vorhanden ist nur lesen
.src	Url desjenigen HTML-Dokumentes, das in dem Layer dargestellt werden soll lesen und schreiben siehe .load()
.top	Y-Koordinate linke obere Layerecke in Pixel, auch negativ String Ziffernfolge eines Integer und "absolute" Pixelangaben bezüglich Fensters "relative" Pixelangaben bezüglich zum umgebenden HTML-Element Hinweis: möglichst nicht kodieren und dafür durch STYLE zu ersetzen: position:absolute;top=.... bzw. position:relative;top=.... lesen und schreiben
.visibility	Sichtbarkeit des Layers unabhängig von der Verschachtelung wenn "show" Layer sichtbar wenn "hide" Layer nicht sichtbar wenn "inherit" Sichtbarkeit laut Eltern lesen und schreiben
.width	Breite des Layers lesen und schreiben
.zIndex	Nummer des Layers in der Folge aller Layer, also auch die in einer Layerverschachtelung Achtung: Die oberste Ebene ist immer das Browserfenster. Damit gilt: Der höchste Wert indiziert den Layer direkt unterhalb des Fensters und somit den obersten Layer aller Layer Der kleinste Wert indiziert den Layer, der am tiefsten in der Folge liegt, also am weitesten entfernt zum Browserfenster dient anstelle von interner Zeigerverwendung per .above und .below Nummernfolge muss nicht im Abstand von 1 sein je kleiner die Nummer um so tiefer in der Folge nur >= 0 (Es kann sein, dass negative Werte toleriert werden. Dann sind Werte < 0 in der Verschachtelungs- hierarchie immer tiefer als 0 oder positiv.) wenn 0 so Browserfenster wenn 1 so der unterste Layer, der am weitesten entfernt ist vom Browserfenster wenn 1000 so der oberste Layer bei Annahme, dass es keine 1001 oder mehr Layer gibt Anzahl der Layer - inklusive verschachtelter Layer - ist nur manuell ermittelbar durch mitzählen (es gibt keine Eigenschaft) lesen und schreiben

Methoden (ausgewählte):

Hinweis: Sämtliche Wertangaben (außer Zeiger) müssen in " " kodiert sein.

.captureEvent(event)	siehe Eventobjekt des Netscape
.handleEvent("event_id")	siehe Eventobjekt des Netscape



<code>.load(url,x)</code>	lädt das HTML-Dokument laut Url in diesen Layer und passt den geladenen Inhalt an die Breite laut x an Achtung: Wenn $x > .clip.width$ $x > width$ so wird der geladene Inhalt abgeschnitten siehe <code>.src</code>
<code>.moveAbove(layer_instanz)</code>	setzt diesen Layer direkt vor den Layer laut <code>layer_instanz</code> und wird damit direkt übergeordneter Layer von <code>layer_instanz</code> ob <code>layer_instanz</code> bereits innerhalb eines Layers liegt, ist egal <code>layer_instanz</code> z.B. laut NAME-Attribut
<code>.moveBelow(layer_instanz)</code>	setzt diesen Layer direkt hinter den Layer laut <code>layer_instanz</code> und wird damit direkt untergeordneter Layer von <code>layer_instanz</code> ob <code>layer_instanz</code> bereits innerhalb eines Layers liegt, ist egal <code>layer_instanz</code> z.B. laut NAME-Attribut
<code>.moveBy(x,y)</code>	Verschiebung des Layers um Pixeldifferenz x und y x in Pixel, auch negativ verändert <code>.left</code> um x y in Pixel, auch negativ verändert <code>.top</code> um y

Beispiel für Fenster des Browser rausschieben und danach neu anzeigen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    var BrowserFenster_AktuelleBreite=2000; // sollte größer als max. übliche Auflösung sein
    var BrowserFenster_AktuelleHoehe=2000; // sollte größer als max. übliche Auflösung sein

    function moveWin()
    {
        for (var i = 1; i < BrowserFenster_AktuelleBreite; i++)
        {window.moveBy(1, 1);}

        window.moveBy((-1)* BrowserFenster_AktuelleBreite,(-1) * BrowserFenster_AktuelleHoehe);
    }
    //-->
</SCRIPT>
</HEAD>
<BODY onload="moveWin();">
</BODY>
</HTML>
```

<code>.moveTo(x,y)</code>	Verschiebung des Layers auf neue Position laut x und y benutzt dabei die Kodierung der Werte von <code>.left</code> und <code>.top</code> : <code>.left</code> bzw. <code>.top</code> können folgende Zusätze haben, die massgebend für die relative Positionierung per <code>moveTo()</code> sind: "absolute" Pixelangaben bezüglich Fensters "relative" Pixelangaben bezüglich zum umgebenden HTML-Element Hinweis: Nullpunkt liegt in der linken oberen Ecke x in Pixel, nur positiv setzt <code>.left</code> auf x y in Pixel, nur positiv setzt <code>.top</code> auf y
<code>.moveToAbsolute(x,y)</code>	Verschiebung des Layers auf neue Position laut x und y immer bezüglich des Browserfensters Hinweis: Nullpunkt liegt in der linken oberen Ecke x in Pixel, nur positiv setzt <code>.pageX</code> auf x y in Pixel, nur positiv setzt <code>.pageY</code> auf y
<code>.releaseEvents(event_typ)</code>	siehe Eventobjekt des Netscape
<code>.resizeBy(x,y)</code>	Änderung der Layerdimension um Pixeldifferenz und ohne Anpassung des Layerinhaltes Achtung: Bei Verkleinerung kann der Inhalt ausgeblendet werden. x in Pixel, auch negativ verändert <code>.width</code> um x y in Pixel, auch negativ



verändert .height um y

`.resizeTo(x,y)` Änderung der Layerdimension auf neue Breite und Höhe und ohne Anpassung des Layerinhaltes
 Achtung: Bei Verkleinerung kann der Inhalt ausgeblendet werden.
 x in Pixel, nur positiv
 setzt .width auf x
 y in Pixel, nur positiv
 setzt .height auf y

`.routeEvent(event_txp)` siehe Eventobjekt des Netscape

Collectionen:

window.document.layers Collection des Netscape (nicht mehr ab Version 6.x)

Feld der Zeiger aller Layer-Objekte im Dokument

Syntax:

`document.layers[index]`

index Integer, ab 0
 muss in [] kodiert sein

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection
 Anzahl **nur der unverschachtelten** Layer
 Hinweis: Anzahl aller - inklusive verschachtelter - Layer nur durch manuelle Indexverwaltung ermittelbar
 (es gibt keine Eigenschaft)
 Layer-Eigenschaft `.zIndex` dient zur Referenzierung auch von verschachtelten Layern

Beispiele:

unverschachtelter Layer:

`<LAYER ID="ID_Layer" STYLE="position: absolute;">Ich bin eine Ebene</LAYER>`

Layer-Zugriff per Script:

`document.ID_Layer`
 oder `document.layers["ID_Layer"]`
 oder `document.layers[0]`

Verwendung eines numerischen Indexes:

Die Indexfolge ist nicht die Folge laut HTML-Kodierung (außer Indexwert 0 = unterster Layer)
 dafür z-index-Attribut kodieren und dieses verwenden

Anzahl der obersten Ebenen auf gleichem Level also ohne deren innere Layer:

`document.layers.length`

verschachtelte Layer:

`<LAYER ID="ebene1" STYLE="position: absolute; left: 100; top: 100;">`
 Nur ein Test...
`<LAYER ID="ebene2" STYLE="position: absolute; left: 50; top: 50;">`
 Jaja!
`</LAYER>`
`</LAYER>`

es wird vererbt: Bsp.: Verschiebung des äußeren Layers (Eltern) verschiebt den inneren (Kind),
 wobei innerer in gleicher relativer Position zum äußeren bleibt wie vor der Verschiebung.

`document.layers.length` liefert 1

Zugriff auf inneren Layer:

`document.ebene1.document.ebene2`

Den Inhalt von Ebenen ändern:

`document.ebene1.document.open();` // anzeigen
`document.ebene1.document.write("Ein neuer Inhalt");` // füllen
`document.ebene1.document.close();` // und schliessen
 // bleibt aber sichtbar

oder `with(document.ebene1.document.ebene2)`
`{`
`open();`
`write("So geht's auch");`
`close();`
`}`

Layer dynamisch erzeugen NACH dem kompletten Laden des Dokumentes

z.B. per `<BODY onload=>`



Achtung:

Ein Versuch Ebenen direkt im HEADER per "<SCRIPT>...</SCRIPT>" zu erzeugen, wird einfach ignoriert !!!!!

Bsp.:

```
var neueEbene=new Layer(breite_in_Pixel);           // instanzieren

neueEbene.document.open();                         // anzeigen
neueEbene.document.write("test");                 // und füllen
neueEbene.document.close();                       // und schliessen
                                                    // bleibt aber sichtbar !!
```

Inhalt eines Layers:

laden Aufruf der Methode "load" oder neueEbene.src="seite1.html";
 anzeigen neueEbene.visibility="show";

verschachtelte Layer dynamisch erzeugen

```
var neueEbene=new Layer(breite_in_Pixel);           // instanzieren
var neueEbene1=new Layer(breite_in_Pixel, document.neueEbene);
```

Eigenschaften von Layer lesen und/oder ändern

z.B. X-Position document.neueEbene.left=200; // Pixel
 Y-Position document.neueEbene.top= 300; // Pixel

Layer verschieben:

Bsp. document.ebene1.moveBy(50, 20);
 oder document.ebene1.left+=50; document.ebene1.top+=20;

4.3.2.1.3.2.6. window.document.link Objekt des Netscape

Erzeugung unter HTML:

Beispiel:

```
<LINK
  HREF="url"                                     // protocol://[host_name[:port]/]dateiname
                                                // protocol://[host_name[:port]/]dateiname#hashtext
                                                // protocol://[host_name[:port]/]dateiname?serachtext
                                                // Bsp: http://www.test.com:8888/test.html
  NAME="anker_name_ohne_#"                     // ist nicht der logische link_name !!
                                                // Netscape + Internet Explorer
oder ID="anker_name_ohne_#"                     // ist der logische link_name !!
                                                // nur Internet Explorer ab 4.x

  TARGET="logischer_window_name"
  onClick="eventhandler1"
  onDbClick="eventhandler2"
  onMouseOut="eventhandler3"
  onMouseOver="eventhandler4"
  onMouseDown="eventhandler5"
  onMouseUp="eventhandler6"
>
  link_text_immer_schreiben
</LINK>
```

Zugriff:

```
document.links[index]

index:      ab 0
           muss in [ ] kodiert sein
```

Beispiel für globalen Style per HEAD:

```
<HEAD>
<STYLE>
  .an {text-decoration: underline overline; color:blue;}
  .aus {text-decoration: none; color:black;}
</STYLE>
</HEAD>
<BODY>
  <A              HREF="test.htm"
                  CLASS="aus"
                  onmouseover="this.className='an';"
                  onmouseout="this.className='aus';"
  >
```



</BODY>

Eigenschaften (ausgewählte):

.className	Klassenreferenz
.hash	entspricht #hashtext zum Anspringen eines Ankers hier den Ankernamen mit vorgesetztem # ablegen
.host	entspricht hostname:port
.hostname	entspricht nur hostname
.href	gesamter Url (kompletter Url) zum Anspringen eines Ankers hier den Ankernamen ohne vorgesetztem # ablegen
.name	entspricht NAME
.offsetLeft	Pixelpos bezüglich linken Rand des HTML-Dokumentes
.pathname	entspricht aus url /dateiname bzw. /dateiname#hashtext bzw. /dateiname?searchtext
.port	lesen und schreiben entspricht port; lesen und schreiben
.protocol	entspricht Protokoll mit Doppelpunkt z.B. "http:" lesen und schreiben
.search	entspricht ?searchtext lesen und schreiben
.target	entspricht TARGET kann auch sein _blank _parent _search _self _top
.text	lesen und schreiben enthält den link_text nur lesen
.x	horizontale Pixelpos gegenüber linke, obere Ecke (0,0) des HTML-Dokumentes nur lesen
.y	vertikale Pixelpos gegenüber linke, obere Ecke (0,0) des HTML-Dokumentes nur lesen

Methoden (ausgewählte):

.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes
.handleEvent()	siehe Eventbehandlung zum NS ab 4.x
.reload([true])	wenn true entfällt: Dokument vom Cache auf Festplatte laden wenn true kodiert: Dokument vom Server und nicht Cache laden Achtung: Server kann selbst Cache haben und daraus das Dokument liefern
.replace(url)	aktuelle Seite mit neuer geladener Seite überschreiben sowie den History-Eintrag zur aktuellen Seite überschreiben (keinen neuen bilden) --> BACK-Button wechselt danach nicht zur überschriebenen Seite, da diese in der History nicht mehr bekannt ist

Collectionen:**window.document.links Collection**

Feld der Zeiger aller Objekte mit HREF-Eigenschaft (Attribut) sowie aller AREA-Objekte im Dokument

Elementefolge laut HTML-Coding

Syntax:

document.links[index]

index Integer, ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

4.3.2.1.3.2.7. window.document.span Objekt des Netscape**Erzeugung in HTML:**

Beispiel: blauesWort

Das DIV- bzw. SPAN-Objekt ähneln sich sehr stark.. SPAN wird als üblicherweise als Textcontainer benutzt.

Die Anzeige (das Rendern) des DIV bzw. SPAN erfolgt erst mit dem Parsen des Ende-Tags, also </DIV> bzw. . Danach können Komponenten zur Laufzeit nachträglich verändert werden, die dann aber nur z.T. sofort sichtbar werden (teilweise erst nach dem Reload des betreffenden Dokumentes).

Vor NS 6.x wird anstelle von DIV- bzw. SPAN-Objekt das LAYER-Objekt favorisiert. Ab NS 6.x wird das Layer-Objekt radikal nicht mehr unterstützt: Es sind für die Ebenendarstellung (überlappende Objekte) keine Layer mehr sondern z.B. DIV oder SPAN verwendbar.



Zugriff:

```
document.ID_Span.eigenschaft
document.ID_Span.methode()
```

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function HandlerFuerOnMoveStart()
    {
        // anstelle des ID vom SPAN falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "green";
        ZeigerAufObjektMitEvent.innerText = "SPAN wird bewegt ";
    }

    function HandlerFuerOnMove ()
    {
        ID_Span1.innerHTML = event.srcElement.offsetLeft;
        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandlerFuerOnMoveEnd()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    // 2-D Positionierung einschalten
    document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offsetTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>
```

4.3.2.1.3.2.8. window.document.style Objekt und window.document.styleSheet Objekt des Netscape (CSS)**Ansatz:**

Aufgrund der HTML-Integration von CSS sollte der Browser CSS unterstützen. Ob diese Unterstützung gegenüber den aktuellen Standards zu HTML und CSS vollständig ist, hängt vom Willen des Browserherstellers ab. Letztendlich wird CSS sinnigerweise als grundlegende Komponente und Eigenschaft **aller** Objekte integriert. CSS bietet elementare Vielfalt bei der dynamischen Scriptprogrammierung, ist gewöhnungsbedürftig, aber durch Normung universell auf diverse Objekte anwendbar, die das STYLE-Attribut bzw. die Eigenschaft .style unterstützen.

Es sei darauf hingewiesen, dass

die Verwendung von CSS-Klassen, Kodierung von <STYLE> bzw. dem STYLE-Attribut bzw. der Eigenschaft .style nur **scheinbar** alle synonym sind:

CSS-Klassen im HEAD können dokumentweit verwendet werden.

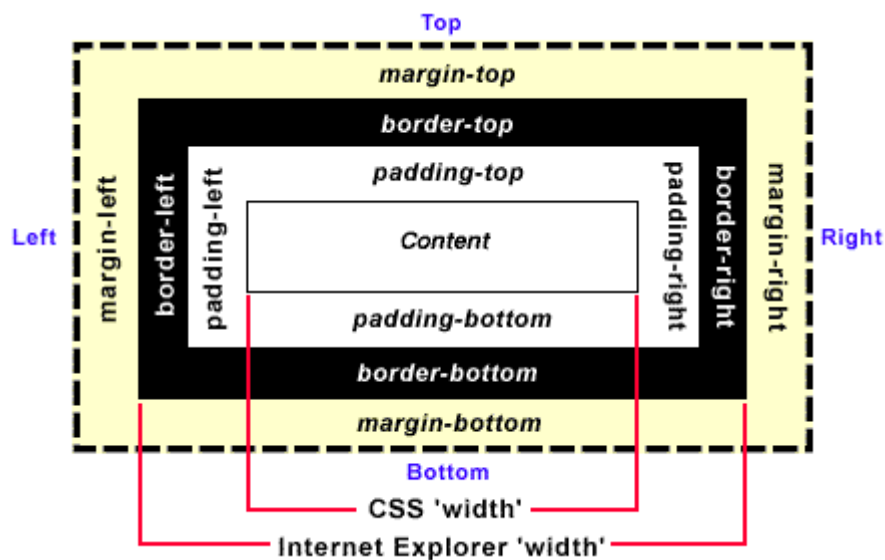
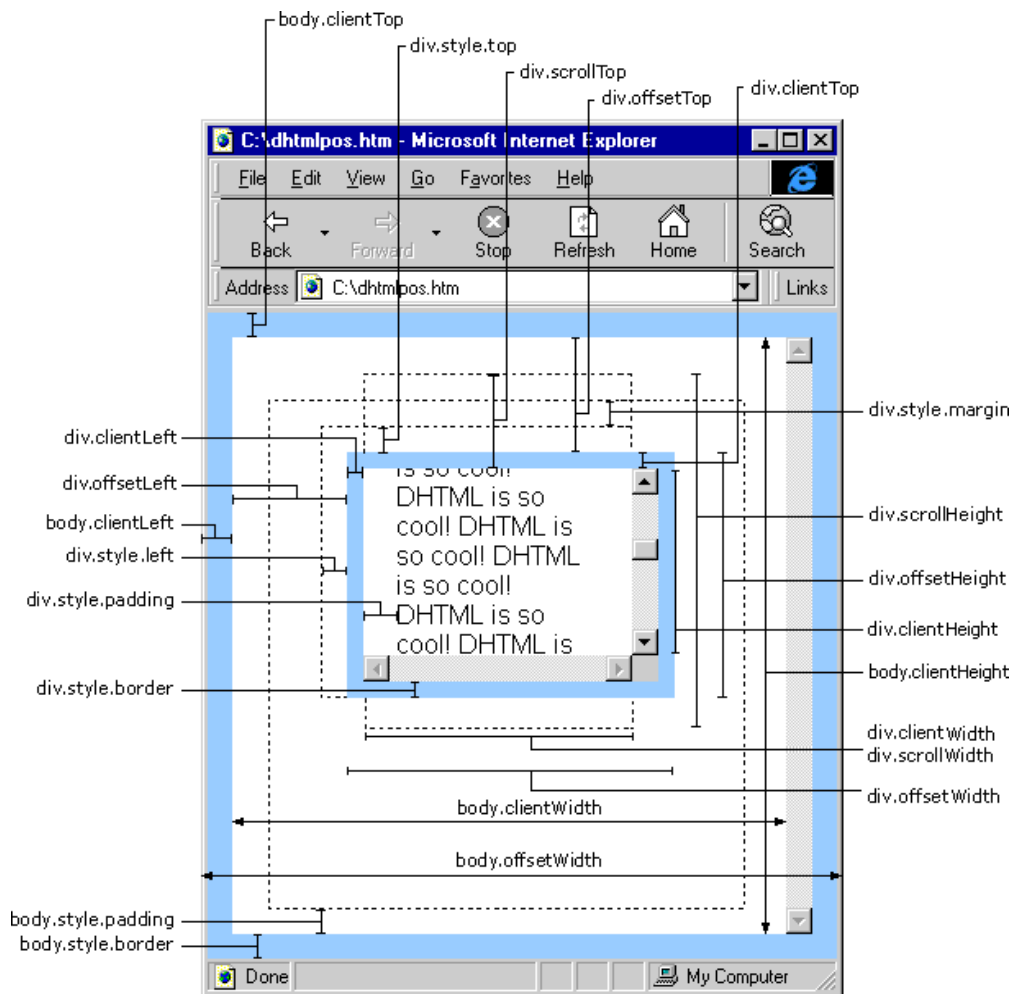
<STYLE>, STYLE-Attribut und .style sind stets HTML-komponentenbezogen.

der Browser diese Formen intern z.T. verschieden (auch bezüglich der Collectionen) verwaltet.

Das Einbinden von Styles aus einer externen Datei ist möglich.



Nachfolgend die Style-Kategorien zum Internet Explorer:



style Objekt und styleSheet Objekt:

Beide Objekte behandeln zwar CSS, unterscheiden sich aber in der Aufgabe:

| | |
|-----------------------|--|
| Das Objekt style | repräsentiert CSS eines Objektes im HTML-Dokument
also z.B. das STYLE-Attribut im HTML-Tag
die Eigenschaft .style
alle Styles sind les- und schreibbar |
| Das Objekt styleSheet | repräsentiert CSS im HTML-DOM, also im gesamten Dokument
wird mit der document.styleSheets Collection verwaltet:
Feld der Zeiger aller styleSheet Objekte im Dokument
Syntax:
<pre>[var ZeigerAufFeld =] document.styleSheets [var ZeigerAufFeldElement =] document.styleSheets [index]</pre> <p>index Integer ab 0
muss in [] kodiert sein</p> <p>Eigenschaft .length Anzahl der Feldelemente
Integer, ab 1</p> |

Nachfolgende Beschreibungen orientieren sich praktischerweise hauptsächlich am Objekt style.

Beispiel: Es sollen DIVs erzeugt werden und deren Objektreferenzen (Zeiger) in einem Zeigerfeld gesammelt werden. Das Zeigerfeld dient der vereinfachten Verwaltung der dynamisch erzeugten DIV's.

```
var DIV_ID="Test_DIV"; // auch "Otto_DIV" etc.möglich , je nach Wunsch

var DIV_Zeiger=Array(); // sammelt alle ID als Zeichenketten

// DIV's dynamisch erzeugen und Zeiger einsammeln
var Left= 20; // Abstand vom linken Fensterrand in Pixeln
for (var i=0; i < 3; i++)
{
    Left+=10; // ab 30 Pixel vom linken Fensterrand
    document.write(      '<DIV ID="' + DIV_ID + i.toString() + "'
                        + ' STYLE="position:absolute;'
                        + 'left=' + Left + 'px;'
                        + 'top=20px;'
                        + '""
                        + '>'
                        + '</DIV>'
    );
    eval ('DIV_Zeiger[ ' + i + ' ] = ' + DIV_ID + i.toString() );
}

// DIV's dynamisch auf dem Bildschirm bezüglich dem linken Fensterrand um 20 Pixel
// verschieben
Left= 40;
for (i=0; i < 3; i++)
{
    Left+=10; // ab 50 Pixel vom linken Fensterrand verschieben
    eval('document.all.DIV_Zeiger[ ' + i + ' ].style.left=' + Left);
}
```

Hinweis: Anstelle von i.toString() kann auch nur i kodiert werden, da der Browser aufgrund von document.write() i automatisch nach String umwandelt. analog für i.toString() innerhalb eval()

Die Verschiebung erfolgt natürlich mit den Objekten, welche innerhalb von <DIV> .. </DIV> kodiert wurden. Im Falle von IMG's innerhalb des DIV werden diese mit verschoben.

Vor allem **wegen** der Manipulation von HTML-Objekten werden DIV's verwendet.

Kodierung der Style-Sheets-Angaben:

innerhalb CSS-Klasse:	immer mit Bindestrich
als .style-Eigenschaft:	ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabe
Bsp:	
in CSS-Klasse	background-color
in Script	.style.backgroundColor

Konvertierungsmethode:

Style besitzt für die style-gerechte Umwandlung eines Url-Wertes die Methode `url()`, welche nicht direkt als Methode des Style-Objektes implementiert ist und damit ohne Punktreferenz kodiert wird.

url(StringWert)	StringWert	Dateiname eines Bildes einer HTC-Datei (siehe Behavior des IE) eines Standard-Behaviors (siehe Behavior des IE) eines Cursors (*.CUR und *.ANI ab IE 6.x) Pfadangabe (relativ oder absolut) mit Dateiname obiger Dateitypen 'none' für nichts (also nicht url() ohne Parameter kodieren !)
Hinweis: alert (zeiger_auf_objekt.style.cursor); liefert beim IE den String 'url(cursor_form)' mit cursor_form für den aktuellen und gesetzten Cursor.		
style.cursor ist standardgemäß mit einer Leerkette belegt, solange kein Cursor gesetzt wird kann nicht mit url(cursor_form) belegt werden, wenn es sich um eine standardgemäß im Browser implementierte Cursorform handelt wie z.B. 'hand' oder 'normal'. Grund: Diese Cursorformen sind keine Dateien (owohl alert den String 'url(cursor_form)' anzeigt).		

Hinweise:

Hinweis zur dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition wird wertmäßig durch den aktuellen Attributwert ersetzt, wenn gleiche Attributnamen / Eigenschaften betroffen sind.

Hinweis zu Style-Begriffen:

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objektinhaltes zum Aussenrand

4.3.2.1.3.2.8.1. Style-Sheet-Deklarationen

Styles innerhalb einer CSS-Klasse immer mit Bindestrich kodieren

Bsp:	in CSS-Klasse	background-color
	in Script	.style.backgroundColor

Style-Sheet-Deklarationen im HEAD (Varianten)

Variante 1 für Pseudoklassen:

```
<HEAD>
  <STYLE TYPE="text/css">
  <!--
      @eigenschaften          /* festkodiert ist @
                              /* z.B. @import
  //-->
  </STYLE>
</HEAD>
```

Variante 2 für CSS-Klasse, die im CLASS-Attribut des HTML-Elementes verwendet wird:

```
<HEAD>
  <STYLE TYPE="text/css">
    <!--
      .freier_klassen_bezeichner {eigenschaften_liste}      /* Punkt ist festkodiert
    oder
      all.freier_klassen_bezeichner {eigenschaften_liste}
    //-->
  </STYLE>
</HEAD>
<BODY>
  <tag_name CLASS="freier_klassen_bezeichner"> ..... </tag_name>
</BODY>
```

. und .all sind synonym

```
freier_klassen_bezeichner:      Bsp:      fettkursiv
```

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

Variante 3 für CSS-Klasse, die im ID-Attribut des HTML-Elementes verwendet wird:

```
<HEAD>
  <STYLE TYPE="text/css">
  <!--
```



```

#freies_id{eigenschaften_liste} /* festkodiert ist #
//-->
</STYLE>
</HEAD>
<BODY>
  <tag_name ID="freies_id"> ..... </tag_name>
</BODY>

freies_id:      Bsp:      fettkursiv
                  innerhalb STYLE mit vorgesetztem # kodieren !
                  innerhalb BODY ohne # kodieren !

eigenschaften_liste: Semikolontrennung
                    muss mit Semikolon enden
                    wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
                    Bsp:      {font-weight:bold;font_style:italic;}

```

Variante 4 für CSS-Klasse, die einem HTML-Tag automatisch und ohne Verwendung des CLASS-Attributes zugeordnet wird, egal wo der Tag im BODY kodiert wurde:

```

<HEAD>
  <STYLE TYPE="text/css">
    <!--
      tag_namen_liste{eigenschaften_liste}
    //-->
  </STYLE>
</HEAD>
<BODY>
  <tag_name_1> ..... </tag_name_1>
  ....
  <tag_name_n> ..... </tag_name_n>
</BODY>

tag_namen_liste:  mindestens 1 Element (Tag-Bezeichner)
                  mehrere Elemente per Komma trennen
                  alle Tags haben die gemeinsame Eigenschaftsliste
                  Gross-klein bei Tag-Bezeichnern egal
                  Bsp:      h1,h2

eigenschaften_liste: Semikolontrennung
                    muss mit Semikolon enden
                    wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
                    Bsp:      {font-weight:bold;font_style:italic;}

```

Variante 5 für CSS-Klasse, die einem HTML-Tag mit Pseudoklasse automatisch und ohne Verwendung des CLASS-Attributes zugeordnet wird, egal wo der Tag im BODY kodiert wurde:

```

<HEAD>
  <STYLE TYPE="text/css">
    <!--
      tag_name:schlüsselwort{eigenschaften_liste} /* Doppelpunkt ist festkodiert
    //-->
  </STYLE>
</HEAD>
<BODY>
  <tag_name> ..... </tag_name>
</BODY>

tag_name:      Schlüsselwort ist Pseudoklasse zum Tag
                  muss kodiert werden
                  Bsp: a:link

eigenschaften_liste: Semikolontrennung
                    muss mit Semikolon enden
                    wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
                    Bsp:      {font-weight:bold;font_style:italic;}

```

Hinweise: Ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das tag-unabhängige ersetzt.
Ein tag-unabhängiges Style-Sheet-Format fügt sich ansonsten zu den tag-abhängigen hinzu.



Variante 6 für CSS-Unterklasse, die im CLASS-Attribut zum entsprechenden HTML-Tag verwendet wird:

```
<HEAD>
  <STYLE TYPE="text/css">
    <!--
      tag_name.unterklasse_name{eigenschaften_liste}      /* Punkt ist festkodiert
    //-->
  </STYLE>
</HEAD>
<BODY>
  <tag_name CLASS="unterklasse_name"> ..... </tag_name>
</BODY>
```

tag_namen.unterklasse_name: Bsp: p.normal, muss aber im CLASS-Attribut zum Tag kodiert werden

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
Bsp: {font-weight:bold;font_style:italic;}

Hinweis: Aufgrund der Kodierung des CLASS-Attributes ist auch Variante 2 verwendbar. Warum Variante 6 existiert, ist unklar.

Beispiele für alle Varianten:

In den Beispielen wurden nur aus Gründen der Übersichtlichkeit Leerzeichen eingefügt.

Beispiel 1

```
<HEAD>
  <STYLE TYPE="text/css">
    <!--
      body{
        background-color:#FFFFFF;
        background-image:url(test.jpg);
        background-repeat:repeat-y;
        margin-top: 1cm;
        margin-left: 2cm;
        margin-right: 1cm;
      }
      p{
        font-family:serif;
        font-size: 12pt;
        text-align:justify;
      }
      p.zusatz_format_p_tag{text-indent:0.5cm;}
      .zusatz_format_alle_tags{
        position:absolute;
        top:4cm;
        z-index:3;
        font-size:40pt;
        font-weight:bold;
        color:blue;
        text-align:center;
      }
    -->
  </STYLE>
</HEAD>
<BODY>
  <SPAN CLASS="zusatz_format_alle_tags"> text </SPAN>
  <P CLASS="zusatz_format_p_tag"> text </P>
</BODY>
```

<!-- --> für Browser kodieren, die CSS nicht können
background-image: Hintergrundbild test.jpg verwenden
Achtung: Möglichst absolute Urls bzw. Pfadangaben verwenden, da
eventuell Browser relative Angaben nicht interpretieren kann
Bsp.: relativ :url('ordner/test.gif')
absolut :url('www.test.de/ordner/test.gif')
background-repeat: repeat-y Hintergrundbild senkrecht wiederholen
repeat-x Hintergrundbild waagerecht wiederholen
no-repeat Hintergrundbild nicht wiederholen
p{ } gilt für JEDES P-Tag ohne CLASS-Attribut zu kodieren
margin-xxxx Seitenrand
text-align Textausrichtung



text-indent	Einzug links
position	Art der position-Angabe folgenden Angaben
Bsp:	position:absolute --> Absolutangaben
top:4cm	hier 4 cm
top	Abstand zum Browserfenster oben
z-index	Angezeigte Schicht (analog zu LAYER)
	--> ab 1 = unterste Schicht
	verwenden für überlappende Textbereiche
p.zusatz_format_p_tag{ ..}	per CLASS-Attribut anwenden im P-Tag

Beispiel 2

```

<HTML>
<HEAD>
  <STYLE TYPE="text/css">
  <!--
      .eigene_css_klasse{ .....}
  -->
  </STYLE>
</HEAD>
<BODY>
  <DIV CLASS=eigene_css_klasse>.....</DIV>
  ....
</BODY>
</HTML>

```

Style-Sheet-Deklaration per STYLE-Attribut im HTML-Element:

alle .style-Eigenschaft ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabenkodieren

Bsp: in CSS-Klasse background-color
 in Script .style.backgroundColor

Beispiel:

```

<BODY>
  <tag_name STYLE=eigenschaften_liste>..... </tag_name>
</BODY>

```

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

Style-Sheet-Deklaration in Script per Eigenschaft .style eines HTML-Elementes:

Beispiel:

```

<DIV ID="ID_Div">Test</DIV>
document.ID_Div.style.height="200px";

```

4.3.2.1.3.2.8.2. Style-Sheet und vordefinierte Werte**Style-Sheet und Farbangaben**

browserspezifisch

Bsp.: "yellow"

Hinweis: frei wählbare Farben per RGB-Angaben im Format

#rrgbbb
 rgb(rr,ggg,bbb)

Rot-Anteil:			
hexa	rr	0 bis FF, mit Vornull	
dezimal 0-255	rrr	ohne Vornullen	
Grün-Anteil:			
hexa	gg	0 bis FF, mit Vornull	
dezimal 0-255	ggg	ohne Vornullen	
Blau-Anteil:			
hexa	bb	0 bis FF, mit Vornull	
dezimal 0-255	bbb	ohne Vornullen	

Beispiele: #FF01A3
 rgb(1,255,10)

Style-Sheet und Dimensionen

pt	Point	=	1/72 inches
pc	Pica	=	12 pt
in	Inch	=	2,54 cm
mm			
cm			



px Pixel

Hinweis: Die kleinste im Browserfenster anzeigbare Pixeleinheit ist 1. Da die Fensterauflösung als Vielfaches von einem **ganzen** Pixel angegeben wird, ist somit z.B. die Pixeleinheit < 1 nicht anzeigbar. Eine numerische Pixeloperation kann in Gleitkomma erfolgen, deren Ergebnis aber zum Rendern als ganze Zahl verwendet wird. Sinnvoll ist also nur die numerische Pixeloperation mit ganzen Zahlen.

Hinweis: Pixel ist nicht in andere Einheiten umrechenbar, da die physische Dimension eines Bildpunktes auf dem Monitor dem eines Pixels entspricht. Welche physische Dimension verwendet wird, hängt vom Monitor ab. Damit lässt sich z.B. die Font-Einheit pt nicht in px umrechnen, es sei denn, man programmiert für genau einen Monitortyp, deren physische Pixeldimension z.B. in cm bekannt ist.
Die Verwendung von px als logische Dimension garantiert aber die exakte Umsetzung in die monitor-spezifische physische Dimension (je nach Bildschirmauflösung). Point wird vorrangig in Layouts verwendet, die nicht pixelorientiert sind, sondern z.B. auf Einheit cm basieren (z.B. Textdarstellung). Alternativen: em, ex, %.

Bei einem Monitor mit 0.25 mm pro Bildpunkt mit einer Auflösung von 1024x768:

100 Pixel sind 3 cm

Buchstabe 'W' mit Schriftart

Arial	fett / nicht fett:	15 pt	sind	4 mm Höhe
				6 mm Breite
Times New Roman	fett / nicht fett	15 pt	sind	3 mm Höhe
				6 mm Breite

em	relativ zur Schrifthöhe des Elementes, das per CSS formatiert wird
ex	relativ zur Höhe des Buchstaben Grosses X
%	Prozentanteil von der Norm des per CSS formatierten Elementes

Style-Sheet- und Dezimalkomma bei numerischen Werten

Dezimalkomma ist der Punkt und NICHT das Komma.

Style-Sheet und Schriften (Font und Style)

vordefinierte Schriftarten:

- serif
- san-serif
- cursive
- fantasy
- monospace

vordefinierte Style:	italic	entspricht kursiv
	oblique	entspricht kursiv
	normal	entspricht nicht kursiv

4.3.2.1.3.2.8.3. Style-Sheet aus Datei einbinden

Style-Sheet-Schriftdatei laden (*.eot bzw. *.prf)

Microsoft: *.eot

Netscape: *.prf

```
<STYLE TYPE="text/css">
<!--
    @font-face{ eigenschaften_liste; src:url (eot_bzw._prf_datei_liste);}
// -->
</STYLE>
```

eigenschaften_liste: optional
Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
Bsp: {font-weight:bold;font_style:italic;}

eot_bzw._prf_datei_liste: mindestens 1 Element
Kommatrennung der Elemente

lokale Fontdateien laden

Fontdateien liegen im FONT-Ordner von Windows

```
<STYLE TYPE="text/css">
<!--
    @font-face{ eigenschaften_liste; local (font_datei_liste);}
oder
    @font-face{ eigenschaften_liste; format (TrueType);}
```



```
// -->
</STYLE>
```

eigenschaften_liste:	optional Semikolontrennung muss mit Semikolon enden wenn Blank enthalten, so Liste in " " bzw. ' ' setzen Bsp: {font-weight:bold;font_style:italic;}
font_datei_liste:	lokale Fontdateien im FONT-Ordner von Windows mindestens 1 Element Blanktrennung der Elemente
TrueType	es werden nur lokale TrueTyp-Fonts (ttf-Dateien) aus dem FONT-Ordner von Windows verwendet

Style-Sheet-Datei laden (*.css)

per LINK-Tag:

```
<LINK
  REL=stylesheet
  TYPE="text/css"
  HREF="url_oder_css_dateiname"
  [MEDIA="typ_name"]
>
<STYLE TYPE="text/css">
  <!-- weitere Stylesheet-Angaben
  // -->
</STYLE>
```

MEDIA=	optional	
	typ_name:	Name des Ausgabe-Mediums
	"all"	alle Medien
	"screen"	Bildschirm
	"print"	Drucker

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x und NS 6.x):

```
<BODY>
  <LINK REL="stylesheet" MEDIA="print" HREF="print.css">
  <DIV CLASS="keindruck">
    alle Seitenteile, die nicht gedruckt werden sollen
  </DIV>
</BODY>
```

print.css enthält nur
.keindruck {display:none; }

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

```
<BODY>
  <LINK REL="stylesheet" MEDIA="screen" HREF="screen.css">
  <DIV CLASS="nurdruck">
    alle Seitenteile, die nur auf Ausdruck zu sehen sein sollen
  </DIV>
</BODY>
```

per Pseudoklasse @import:

```
<STYLE TYPE="text/css">
  <!--
    @import (url_oder_css_datei) typ_liste;
  // -->
</STYLE>
```

typ_liste: Liste der Ausgabe-Medien
Kommatrennung
Ausgabe-Mediums können sein



"all"	alle Medien
"screen"	Bildschirm
"print"	Drucker

Bsp: print,screen;

4.3.2.1.3.2.8.4. Style-Sheet und Wertzuweisung an Eigenschaften

eigenschaft: wert; oder eigenschaft=wert;

Hinweis: Pflichtkodierung von Doppelpunkt bzw. Gleichheitszeichen
Semikolon

wenn für wert auto kodiert werden kann, so ist das der Standardwert, falls Style nicht belegt wird

Folge von Eigenschaften möglich: als Liste mit Semikolontrennung

Bsp: eigenschaft1=wert1;; eigenschaft_n=wert_n;

wenn Blank in Kodierung enthalten, so alles in " " bzw. ' ' setzen

Bsp: style="font-style: italic; color:red;"

wenn Eigenschaft mit mehreren Werten belegbar, so

Werte durch Blank trennen

gesamte Liste in " " bzw. ' ' kodieren wegen den Trennblanks

Bsp: "background: url(test.gif) no-repeat middle;"

4.3.2.1.3.2.8.5. Style-Sheet und Ausgabemedien (Pseudoklasse @media)

<STYLE TYPE="text/css">

<!--

@media typ_liste{eigenschaften_liste}

// -->

</STYLE>

typ_liste: Liste der Ausgabe-Medien
Kommatrennung
Ausgabe-Mediums können sein

"all"	alle Medien
"screen"	Bildschirm
"print"	Drucker

Bsp: print,screen;

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
Bsp: {font-weight:bold;font_style:italic;}**4.3.2.1.3.2.8.6. Style-Sheet und Seiten-Eigenschaften (Pseudoklasse (@page))**

@page:first {eigenschaften_liste}	Regel gehört der 1. Seite
@page:footer {eigenschaften_liste}	Regel gehört der Fußnote
@page:header {eigenschaften_liste}	Regel gehört der Kopfnote
@page:left {eigenschaften_liste}	Regel gehört der linken Seite
@page:left:header {eigenschaften_liste}	Regel gehört der Kopfnote Seite links
@page:left:footer {eigenschaften_liste}	Regel gehört der Fußnote Seite links
@page:right {eigenschaften_liste}	Regel gehört der rechten Seite
@page:right:header {eigenschaften_liste}	Regel gehört der Kopfnote Seite rechts
@page:right:footer {eigenschaften_liste}	Regel gehört der Fußnote Seite rechts

Beispiel: @page : left {font-weight:bold;font_style:italic;}

4.3.2.1.3.2.8.7. Style-Sheet und HTML-Tag-bezogene Eigenschaften

Hinweis: Ist für einen Style der Wert auto kodierbar, so ist das der Standardwert, falls Style nicht belegt wird.

Style-Sheet-Eigenschaften zu <A> und seine Pseudoklassen

a:link {eigenschaften_liste}	Eigenschaften für noch nicht besuchte Links
a:visited {eigenschaften_liste}	Eigenschaften für schon besuchte Links
a:active {eigenschaften_liste}	Eigenschaften für gerade besuchtes Link
a:hover {eigenschaften_liste}	Eigenschaften für Maus über Link , ab IE 4.x nur für Tags mit HREF-Attribut

Style-Sheet-Eigenschaften zu <P> und seine Pseudoklassen

p:first-line {eigenschaften_liste}	Eigenschaften Absatz 1. Zeile
p:first-letter {eigenschaften_liste}	Eigenschaften Absatz 1. Zeile, 1. Zeichen
p:before {content:"freier_text" }	Text vor dem Element einfügen



p:after{content:"freier_text" } Text nach dem Element einfügen

Style-Sheet-Eigenschaften zu <H1> bis <H6>

h1:first-line{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile
 h1:first-letter{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile, 1. Zeichen

für H2 bis H6 analog

Beim Internet Explorer 6.0 unter Windows 98 bzw. Windows XP wurde die Darstellung der Überschriften verändert, wenn der jeweilige Standardfont benutzt wird. Empfehlung: Neudefinition der <Hx>-Tags per Style (soweit möglich).

Style-Sheet-Eigenschaften für Tabelle

caption-side:top; Überschrift oberhalb zentriert
 caption-side:topleft; Überschrift oberhalb linksbündig
 caption-side:topright; Überschrift oberhalb rechtsbündig
 caption-side:bottom; Überschrift unterhalb zentriert
 caption-side:bottomleft; Überschrift unterhalb linksbündig
 caption-side:bottomright; Überschrift unterhalb rechtsbündig

row-span: anzahl_der_zeilen_über_die_sich_die_zelle_ausstrecken_soll;
 column-span: anzahl_der_spalten_über_die_sich_die_zelle_ausstrecken_soll;

Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet)

display: list-item; Element in eigenem Absatz UND mit Bullet anzeigen

Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung)

list-style-type:decimal; 1 2 3 ...
 list-style-type:lower-roman; i ii iii ..
 list-style-type:lower-alpha a b c
 list-style-type:upper-roman; I II III ...
 list-style-type:upper-alpha A B C
 list-style-type:disk; Bullet ist Dateisymbol
 list-style-type:circle; Bullet ist rund
 list-style-type:square; Bullet ist rechteckig
 list-style-type:none;

list-style-position:inside; Listenelement einrücken; ist Standard
 list-style-position:outside; Listenelement ausrücken

list-style-image: url(bullet_grafik_datei); GIF oder JPG

list-style: liste_aller_obigen_eigenschaften_mit_blank_trennung;

Style-Sheet und Dimension von Elementen

height:wert; Höhe des Objektes
 wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 Prozentangabe
 Hinweis: wenn geändert wird, so kann automatischer Umbruch des Inhaltes des HTML-Elementes erfolgen

height:auto;

width:wert; Breite des Objektes
 wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 Prozentangabe
 Hinweis: wenn geändert wird, so kann automatischer Umbruch des Inhaltes des HTML-Elementes erfolgen

width:auto;

Style-Sheet und Position von Elementen

position: absolute; Positionsangaben bezüglich Anzeigefenster-Rand, Element ist scrollbar
 position: fixed; Positionsangaben bezüglich Anzeigefenster-Rand, Element ist nicht scrollbar
 position: relative; Positionsangaben bezüglich Vorgänger-Element
 position: static; hebt absolute bzw. fixed bzw. relative auf

top:wert; Abstand zum oben umgebenden Objekt ab IE 5.x
 wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 Prozentangabe

top:auto;

bottom:wert; Abstand zum unten umgebenden Objekt ab IE 5.x
 wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 Prozentangabe



bottom:auto;	
left:wert;	Abstand zum links umgebenden Objekt ab IE 5.x wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
left:auto;	
right:wert;	Abstand zum rechts umgebenden Objekt ab IE 5.x wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
right:auto;	
offset-left:	wie left ab IE 5.x
offse-top:	wie top ab IE 5.x
pixel-left:wert;	absolute X-Position der linken oberen Objekt-Ecke im Pixelsystem, dessen Ursprung (0,0) links oben liegt, also Abstand vom linken Fensterrand wert nur Integer und ohne Einheit, da automatisch in Pixel interpretiert
pixel-top:wert;	absolute Y-Position der linken oberen Objekt-Ecke im Pixelsystem, dessen Ursprung (0,0) links oben liegt, also Abstand vom oberen Fensterrand wert nur Integer und ohne Einheit, da automatisch in Pixel interpretiert

Style-Sheet und Abstand von HTML-Elementen

top:abstand_obenhalb_in_pixel;	
top:auto;	
left:abstand_links_in_pixel;	
left:auto;	
bottom:abstand_unterhalb_in_pixel;	
bottom:auto;	
right:abstand_rechts_in_pixel;	
right:auto;	
margin: werte_liste_von_breiten_mit_blank_trennung;	Randbreite aller 4 Seiten wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten 2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für links UND rechts 3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für rechts 3. Wert gilt für unten 4. Wert gilt für links
margin:auto;	
margin-top: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite oben wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-top:auto;	
margin-bottom: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite unten wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-bottom:auto;	
margin-left: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite links wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-left:auto;	
margin-right: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite rechts wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-right:auto;	
padding: werte_liste_von_breiten_mit_blank_trennung;	Innenabstand aller 4 Seiten



wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten 2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für links UND rechts 3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für rechts 3. Wert gilt für unten 4. Wert gilt für links
padding:auto; padding-top: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand oben wert ist Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-top:auto; padding-bottom: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand unten wert ist Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-bottom:auto; padding-left: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand links wert ist Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-left:auto; padding-right: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand rechts wert ist Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-right:auto;	

Style-Sheet und Überlagerung von Elementen

z-index:auto;	jedes neu erzeugte Objekt ist das oberste
	Achtung: Sollte die Dimension des neuen Objektes sich nach der Erzeugung derart verändern, so dass es ein anderes Objekt überlagert, dann ändert sich trotzdem nichts an der Sichtbarkeit
z-index:wert;	Schichtnummer des Objektes bei sich irgendwann überlagernden Objekten (also auch bei Überlagerung, die erst nach deren Erzeugung erfolgt)
Wert	Integer ganzzahlig, also auch < 0 je kleiner um so tiefer in der Schicht je höher um so höher in der Schicht Sichtbar ist immer die oberste, also höchste Schicht wenn 2 Objekte mit identischer Schichtnummer, so Sichtbarkeit laut Reihenfolge der Kodierung im Quelltext

Style-Sheet und HTML-Elemente-Anordnung im Dokument

direction:ltr;	von links nach rechts; ist Standard
direction:rtl;	von rechts nach links
display:block;	Element in eigenem Absatz anzeigen
display:inline;	Element nicht in eigenem Absatz anzeigen
display:none;	Element nicht anzeigen
display:list-item;	
float:left;	Element linksbündig; Textfluss rechts
float:right;	Element rechtsbündig; Textfluss links
float:none;	Standard
clear:left;	hebt float:left; auf
clear:right;	hebt float:right; auf
clear:both;	hebt float:left; UND float:right; auf
clear:none;	identisch mit float:none;

Style-Sheet und HTML-Element-Dimension (-Grenzen, -Anzeigebereich)

width: breite_in_pixel;
width:auto;
min-width: minimale_breite_in_pixel;



min-width:auto;
 max-width: maximale_breite_in_pixel;
 max-width:auto;

height: hoehe_in_pixel;
 height:auto;
 min-height: minimale_hoehe_in_pixel;
 min-height:auto;
 max-height: maximale_hoehe_in_pixel;
 max-height:auto;

overflow:hidden; wenn Elementgröße > max-width und max-height, so wird Element auf Maximalwerte begrenzt
 overflow:visible; wenn Elementgröße > max-width und max-height, so werden Maximalwerte ignoriert

Style-Sheet und HTML-Element-Sichtbarkeit

visibility:hidden; unsichtbar, aber Platzhalter vorhanden, so dass das HTML-Element im Layout unsichtbar bleibt
 visibility:visible; sichtbar
 visibility:inherit; Sichtbarkeit laut Elternobjekt

display:none; unsichtbar UND kein Platzhalter möglich, also HTML-Element auch aus dem Layout entfernen
 display:block; sichtbar
 display:inline; sichtbar

Style-Sheet und Hintergrund-Grafik

background-image:url (grafik_datei); Hintergrunddatei GIF oder JPG
 background-image:none;

background-color:#rrggb; Hintergrundfarbe
 background-color:rgb(rrr,ggg,bbb);
 background-color:vordefinierte_farbe;

background-repeat:repeat; Hintergrundgrafik kacheln auf gesamten Hintergrund
 background-repeat:repeat-x; Hintergrundgrafik für 1 Zeile kacheln
 background-repeat:repeat-y; Hintergrundgrafik für 1 Spalte kacheln
 background-repeat:no-repeat; kein Kacheln des Hintergrundbildes

background-attachment:scroll; Hintergrundgrafik scrollt mit Vordergrund
 background-attachment:fixed; Hintergrundgrafik scrollt nie

background-position:top; Hintergrundgrafik auf Oberkante Hintergrund
 background-position:center; Hintergrundgrafik zentriert auf Hintergrund
 background-position:middle; Hintergrundgrafik mittig auf Hintergrund
 background-position:bottom; Hintergrundgrafik auf Unterkante Hintergrund
 background-position:left; Hintergrundgrafik linksbündig auf Hintergrund
 background-position:right; Hintergrundgrafik rechtsbündig auf Hintergrund

background: eigenschaften_liste_mit_blank_trennung; alle obigen Eigenschaften kodierbar mit Blanktrennung

Bsp: background: url(back.gif) no-repeat middle; Blank-Trennung !!

-moz-opacity:faktor Transparenz des Hintergrundbildes
 nur NS 6.x
 Faktor > 0 und bis 1
 Bsp: 0.9 entspricht 90%
 Bsp:

Style-Sheet und Rahmen-Eigenschaften (Border)

border-color:#rrggb; Rahmenfarbe aller 4 Seiten
 border-color:rgb(rrr,ggg,bbb);
 border-color:vordefinierte_farbe;
 border-top-color:#rrggb; Rahmenfarbe oben
 border-top-color:rgb(rrr,ggg,bbb);
 border-top-color:vordefinierte_farbe;
 border-bottom-color:#rrggb; Rahmenfarbe unten
 border-bottom-color:rgb(rrr,ggg,bbb);
 border-bottom-color:vordefinierte_farbe;
 border-left-color:#rrggb; Rahmenfarbe links
 border-left-color:rgb(rrr,ggg,bbb);
 border-left-color:vordefinierte_farbe;
 border-right-color:#rrggb; Rahmenfarbe rechts
 border-right-color:rgb(rrr,ggg,bbb);



border-right-color:vordefinierte_farbe;

border-style:none;
border-style:dotted;
border-style:dashed;
border-style:solid;
border-style:double;
border-style:groove;
border-style:ridge;
border-style:inset;
border-style:outset;
border-top-style:none;
border-top-style:dotted;
border-top-style:dashed;
border-top-style:solid;
border-top-style:double;
border-top-style:groove;
border-top-style:ridge;
border-top-style:inset;
border-top-style:outset;
border-bottom-style:none;
border-bottom-style:dotted;
border-bottom-style:dashed;
border-bottom-style:solid;
border-bottom-style:double;
border-bottom-style:groove;
border-bottom-style:ridge;
border-bottom-style:inset;
border-bottom-style:outset;
border-left-style:none;
border-left-style:dotted;
border-left-style:dashed;
border-left-style:solid;
border-left-style:double;
border-left-style:groove;
border-left-style:ridge;
border-left-style:inset;
border-left-style:outset;
border-right-style:none;
border-right-style:dotted;
border-right-style:dashed;
border-right-style:solid;
border-right-style:double;
border-right-style:groove;
border-right-style:ridge;
border-right-style:inset;
border-right-style:outset;

border-width:wert

border-width:medium
border-width:thin
border-width:thick
border-top-width:wert
oder cm
border-top-width:medium
border-top-width:thin
border-top-width:thick
border-bottom-width:wert
border-bottom-width:medium
border-bottom-width:thin
border-bottom-width:thick
border-left-width:wert
oder cm
border-left-width:medium
border-left-width:thin
border-left-width:thick
border-right-width:wert
border-right-width:medium
border-right-width:thin
border-right-width:thick

kein Rahmen um das gesamte Element
gepunkteter Rahmen um das gesamte Element
gestrichelter Rahmen um das gesamte Element
einfacher durchgezogener Rahmen um das gesamte Element
doppelter durchgezogener Rahmen um das gesamte Element
3D-Effekt 1 um das gesamte Element
3D-Effekt 2 um das gesamte Element
3D-Effekt 3 um das gesamte Element
3D-Effekt 4 um das gesamte Element
kein Rahmen oben
gepunkteter Rahmen oben
gestrichelter Rahmen oben
einfacher durchgezogener Rahmen oben
doppelter durchgezogener Rahmen oben
3D-Effekt 1 oben
3D-Effekt 2 oben
3D-Effekt 3 oben
3D-Effekt 4 oben
kein Rahmen unten
gepunkteter Rahmen unten
gestrichelter Rahmen unten
einfacher durchgezogener Rahmen unten
doppelter durchgezogener Rahmen unten
3D-Effekt 1 unten
3D-Effekt 2 unten
3D-Effekt 3 unten
3D-Effekt 4 unten
kein Rahmen links
gepunkteter Rahmen links
gestrichelter Rahmen links
einfacher durchgezogener Rahmen links
doppelter durchgezogener Rahmen links
3D-Effekt 1 links
3D-Effekt 2 links
3D-Effekt 3 links
3D-Effekt 4 links
kein Rahmen rechts
gepunkteter Rahmen rechts
gestrichelter Rahmen rechts
einfacher durchgezogener Rahmen rechts
doppelter durchgezogener Rahmen rechts
3D-Effekt 1 rechts
3D-Effekt 2 rechts
3D-Effekt 3 rechts
3D-Effekt 4 rechts

Rahmendicke aller 4 Seiten

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm

mittel

dünn

dick

Rahmendicke oben, wert ist Fließkommazahl mit Einheitenangabe z.B. pt

mittel oben

dünn oben

dick oben

Rahmendicke unten, wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm

mittel unten

dünn unten

dick unten

Rahmendicke links, wert ist Fließkommazahl mit Einheitenangabe z.B. pt

mittel links

dünn links

dick links

Rahmendicke rechts, wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm

mittel rechts

dünn rechts

dick rechts



border-top:eigenschaften_liste;
border-bottom: eigenschaften_liste;
border-right: eigenschaften_liste;
border-left: eigenschaften_liste;
border:eigenschaften_liste;

passende Werte-Liste für Rahmen oben, Werte mit Blank trennen
passende Werte-Liste für Rahmen unten, Werte mit Blank trennen
passende Werte-Liste für Rahmen rechts, Werte mit Blank trennen
passende Werte-Liste für Rahmen links, Werte mit Blank trennen
Werte-Liste **aller** möglichen Bordereigenschaften, Werte mit Blank trennen

Bsp: border-top: thick inset rgb(192,192,255);

Style-Sheet und Element-Ausschnitt

clip: rect (x1,y1,x2,y2);

Ausschnitt

Pixelangaben bezüglich Elementgrenze

Form des Bildausschnittes: RECHTECKIG

wird eigentlich nur zum Schreiben verwendet

x1,y1 linke obere Ecke

x1

horizontal in Pixel

y1

vertikal in Pixel

x2,y2 rechte untere Ecke

x2

horizontal in Pixel

y2

vertikal in Pixel

x1 bis y2 immer bezüglich links oben im HTML-Element

x1,y1, x2,y2 können den Wert auto haben, der immer die maximale Dimension bewirkt

Bsp. 'rect(auto,y1,auto,y2)';

wenn alle auf auto, so wird der Bildausschnitt maximiert, was der

Standarddimension des Elementes entspricht (kein Ausschnitt)

clip:auto;

kein Ausschnitt

entspricht clip:rect(auto,auto,auto,auto);

Style-Sheet und Element-Scrolling

Scrolling nötig, wenn

Elementgröße > max-width und max-height
aber Maximalwerte eingehalten werden sollen

overflow:scroll;

Scrolling immer möglich

overflow:auto;

siehe auch overflow:visible; bzw. overflow:hidden;

Style-Sheet-Eigenschaften für Seitendarstellung im Dokument

margin: werte_liste_von_breiten_mit_blank_trennung;

Randbreite aller 4 Seiten

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
Prozentangabe z.B. 3%

wenn nur 1 Wert kodiert: gilt für oben UND unten UND links UND rechts

wenn 2 Werte kodiert: 1. Wert gilt für oben UND unten
2. Wert gilt für links UND rechts

wenn 3 Werte kodiert: 1. Wert gilt für oben
2. Wert gilt für links UND rechts
3. Wert gilt für unten

wenn 4 Werte kodiert: 1. Wert gilt für oben
2. Wert gilt für rechts
3. Wert gilt für unten
4. Wert gilt für links

margin:auto;

margin-top: werte_liste_obiger_breiten_mit_blank_trennung;

Randbreite oben

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
Prozentangabe z.B. 3%

margin-top:auto;

margin-bottom: werte_liste_obiger_breiten_mit_blank_trennung;

Randbreite unten

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
Prozentangabe z.B. 3%

margin-bottom:auto;

margin-left: werte_liste_obiger_breiten_mit_blank_trennung;

Randbreite links

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
Prozentangabe z.B. 3%

margin-left:auto;



margin-right: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite rechts wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-right:auto;	
size:seiten_breite;	z.B. 20cm
size:seiten_hoehe;	z.B. 29cm
size:seiten_breite seiten_hoehe;	Blanktrennung der beiden Werte
size:landscape;	Querformat
size:portrait;	Hochformat
size:auto;	
page-break_before:always;	immer Seitenumbruch vor aktuellem Element erzeugen
page-break_before:aroid;	nie Seitenumbruch vor aktuellem Element erzeugen
page-break_before:left;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann linksbündig ablegen
page-break_before:right;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann rechtsbündig ablegen
page-break_before:auto;	
page-break_after:always;	immer Seitenumbruch nach aktuellem Element erzeugen
page-break_after:aroid;	nie Seitenumbruch nach aktuellem Element erzeugen
page-break_after:left;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element linksbündig ablegen
page-break_after:right;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element rechtsbündig ablegen
page-break_after:auto;	

Style-Sheet-Eigenschaften für Text

text-indent:wert;	Texteinschub wert ist Fließkommazahl mit Einheit z.B. mm oder cm oder pt Prozent z.B. 3% wert > 0, so Textzeile einrücken wert < 0, so Textzeile ausrücken
text-align:left;	Text linksbündig
text-align:center;	Text zentriert
text-align:right;	Text rechtsbündig
text-align:justify;	Text im Blocksatz
vertical-align:top;	Text auf Oberkante der Umgebung
vertical-align:middle;	Text auf Mitte der Umgebung
vertical-align:bottom;	Text auf Unterkante der Umgebung
vertical-align:sub;	Text tieferstellen
vertical-align:super;	Text höher stellen
vertical-align:baseline;	Text auf Basislinie des umgebenden Textes mit verschiedenen Schriftarten
vertical-align:text-top;	Text auf obere Linie des umgebenden Textes mit verschiedenen Schriftarten
vertical-align:text-bottom;	Text auf untere Linie des umgebenden Textes mit verschiedenen Schriftarten
font-size:schriftgroesse_wert;	z.B. 2pt oder vordefiniert
line-height:zeilenhoehe_wert;	z.B. 2pt* oder vordefiniert
white-space:normal;	automatischer Zeilenumbruch einschalten, ab IE 5.x
white-space:pre;	manuellen Zeilenumbruch einschalten, ab IE 5.x
white-space:nowrap;	kein Zeilenumbruch möglich, ab IE 5.x
color:#rrggbb;	
color:rgb(rrr,ggg,bbb);	
color:vordefinierte_farbe;	
columns:anzahl_der_text_spalten_fuer_textfluss;	
column-gap:abstand_der_text_spalten_fuer_textfluss;	
column-rule-width:dicke_des_trennstriches_zwischen_den_textspalten_vom_textfluss;	
column-rule-color:#rrggbb; oder rgb(rrr,ggg,bbb);	Farbe Trennstrich
column-rule-style:none;	kein Trennstrich zwischen Textspalten
column-rule-style:dotted;	gepunkteter Trennstrich
column-rule-style:dashed;	gestrichelter Trennstrich
column-rule-style:solid;	einfacher durchgezogener Trennstrich
column-rule-style:double;	doppelter durchgezogener Trennstrich
column-rule-style:groove;	3D-Effekt 1
column-rule-style:ridge;	3D-Effekt 2
column-rule-style:inset;	3D-Effekt 3



column-rule-style:outset;	3D-Effekt 4
column-rule:werte_liste_aus_obigen_textfluss_werten_mit_blank_trennung;	
word-spacing: abstand_zwischen_woertern_im_text;	
word-wrap:normal;	kein Wortumbruch, ab IE 5.x
word-wrap:break-word;	Wortumbruch, ab IE 5.x
letter-spacing:abstand_zwischen_zeichen_im_text;	Fliesskommazahl mit Einheit z.B. mm oder cm oder pt
letter-spacing:normal;	
line-height:abstand_zwischen_2_zeilen;	Fliesskommazahl mit Einheit z.B. mm oder cm oder pt
line-height:normal;	
text-decoration:underline;	Text unterstrichen
text-decoration:overline;	Text überstrichen
text-decoration:line-through;	Text durchgestrichen
text-decoration:blink;	Text blinkend
text-decoration:none;	Text normal
text-transform:capitalize;	Wortanfang ALLER Wörter auf Großbuchstabe
text-transform:uppercase;	alle Zeichen nach Großbuchstabe
text-transform:lowercase;	alle Zeichen nach Kleinbuchstabe
text-transform:none;	Text normal
text-shadow:#rrggb;	
text-shadow:rgb(rrr,ggg,bbb);	
text-shadow:vordefinierte_farbe;	
text-shadow:none;	kein Textschatten
orphans:anzahl_der_VOR_seitenumbruch_zusammenzuhaltender_zeilen;	Standard ist 2 entspricht Schusterjunge
widow:anzahl_der_NACH_seitenumbruch_zusammenzuhaltender_zeilen;	Standard ist 2 entspricht Hurenkind

Style-Sheet-Eigenschaften für Unicode (Zeichensatz)

unicode-range:U+xxxx-yyyy;

Fragezeichen als Joker innerhalb von xxxx und yyyy verwendbar

Bsp: unicode-range:U+0000-007F; ist ASCII-Zeichensatz
 unicode-range:U+0000-00?F; ? steht für 0 bis F --> ist nicht ASCII-Zeichensatz

Style-Sheet-Eigenschaften für Font

font-family: schriftarten_liste_mit_kommatrennung;

es wird **nur** der **ERSTE** auf dem lokalen Rechner **gefundene** Font aus der Liste geladen
 wenn Blank vorhanden, so alles in " " bzw. ' ' setzen

vordefinierte Schriftarten: serif
 san-serif
 cursive
 fantasy
 monospace

font-style:italic; kursiv
 font-style:oblique; kursiv
 font-style:normal; nicht kursiv

font-variant:small-caps; kleine Großbuchstaben (Kapitälchen)
 font-variant:normal; kein Kapitälchen

font-size:schrift_groesse_wert; Fliesskommazahl mit Einheit z.B. mm oder cm oder pt
 Prozent z.B. 3%

font-size:xx-small; winzig
 font-size:x-small; sehr klein
 font-size:small; klein
 font-size:medium; mittel
 font-size:large; groß
 font-size:x-large; sehr groß
 font-size:xx-large; riesig
 font-size:smaller; etwas kleiner als normal



font-size:larger;	etwas größer als normal
font-weight:bold;	fett
font-weight:bolder;	extra fett
font-weight:lighter;	dünn
font-weight:normal;	nicht dünn und nicht fett
font-weight:100;	extra dünn
font-weight:200;	
font-weight:300;	
font-weight:400;	
font-weight:500;	medium
font-weight:600;	
font-weight:700;	bold
font-weight:800;	
font-weight:900;	extra fett

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;

Liste darf maximal 6 der nachfolgenden Eigenschaften beinhalten

caption	Zeichensatz für Objekte mit Überschriften
font-style	
font-variant	
font-weight	
font-size	
line-height	
font-family	
icon	Zeichensatz für Grafik
menu	Zeichensatz in Menüs
message-box	Zeichensatz in Messages-Boxen
small-caption	Zeichensatz für Control-Elemente
status-bar	Zeichensatz für Statuszeile

Style-Sheet-Eigenschaften für Mauscursor

cursor:auto;	
cursor:default;	je nach Windowseinstellung
cursor:hand;	
cursor:crosshair;	Fadenkreuz
cursor:pointer;	Zeiger
cursor:move;	Kreuz für Beweglichkeit
cursor:n-resize;	Pfeil Nord
cursor:ne-resize;	Pfeil Nord-Ost
cursor:nw-resize;	Pfeil Nord-West
cursor:e-resize;	Pfeil Ost
cursor:se-resize;	Pfeil Süd-Ost
cursor:s-resize;	Pfeil Süd
cursor:sw-resize;	Pfeil Süd-West
cursor:w-resize;	Pfeil West
cursor:text;	also senkrechter Strich
cursor:wait;	Sanduhr
cursor:help;	Fragezeichen
cursor:url(url_oder_mauscursor_grafik_datei);	GIF oder JPG

Style-Sheet-Eigenschaften für Scrollbalken (Scrollbars)

scrollbar3d-light-color:#rrgbbb	Scrollbar-Farbe bei 3D
scrollbar3d-light-color:rgb(rrr,ggg,bbb);	
scrollbar3d-light-color:vordefinierte_farbe;	
scrollbar-arrow-color:#rrgbbb;	Scrollbar-Pfeile-Farbe ab IE 5.x
scrollbar-arrow-color:rgb(rrr,ggg,bbb);	
scrollbar-arrow-color:vordefinierte_farbe;	
scrollbar-base-color:#rrgbbb;	Scrollbar-Basis-Farbe ab IE 5.x
scrollbar-base-color:rgb(rrr,ggg,bbb);	
scrollbar-base-color:vordefinierte_farbe;	
scrollbar-dark-shadow-color:#rrgbbb;	Scrollbar-Farbe des dunklen Schattens ab IE 5.x
scrollbar-dark-shadow-color:rgb(rrr,ggg,bbb);	
scrollbar-dark-shadow-color:vordefinierte_farbe;	
scrollbar-face-color:#rrgbbb;	Scrollbar-Face-Farbe ab IE 5.x
scrollbar-face-color:rgb(rrr,ggg,bbb);	
scrollbar-face-color:vordefinierte_farbe;	
scrollbar-highlight-color:#rrgbbb;	Scrollbar-Highlight-Farbe ab IE 5.x
scrollbar-highlight-color:rgb(rrr,ggg,bbb);	




```

scrollbar-highlight-color:vordefinierte_farbe;
scrollbar-shadow-color:#rrggbb;           Scrollbar-Farbe des Schattens ab IE 5.x
scrollbar-shadow-color:rgb(rr,ggg,bbb);
scrollbar-shadow-color:vordefinierte_farbe;

```

Style-Sheet-Eigenschaften für Zoom eines Elementes

```

zoom:wert;                               Objekt vergrößern ab IE 5.x
wert ist   Fließkommazahl als Faktor (1,0 ist normal)
           Prozentangabe mit % z.B. 50% (100% ist normal)

zoom:normal;

```

4.3.2.1.3.2.8.8. Style-Sheet-Beispiele**Beispiel 1**

```

<STYLE TYPE="text/css">
<!--
    h1 { font-size:24pt;margin-top:1.2cm;margin-left:30px;}
    body { background-color:rgb(51,0,102);}
    p,li { font-size:12pt;line-height:14pt;font-family:Helvetica,Arial;}
    p.normal { fonszie:10pt;color:bllack;}
    p.klein { fontsize:8pt;color:black}
    all.rot { color:red;}   oder .rot { color:red}
                                Rot-Definition für ALLE Tags
                                Anwendung z.B. per <P CLASS="normal">text</P>
                                Anwendung z.B. per <P CLASS="rot">text</P>
                                Anwendung z.B. per <H1 CLASS="rot">text</H1>
    #fett_kursiv { font-weight:bold;font-style:italic;}
                                Anwendung z.B. per <P ID="fett_kursiv">text</P>
    a:link { color:#FF0000;font-weight:bold;}
                                Anwendung z.B. per <A HREF="....." >text</A>

// -->
</STYLE>

```

Anwendung je nach Tag --> meist innerhalb <BODY>

Beispiel 2

```

<BODY>
    <DIV STYLE="background-color:#FF0000;">test</DIV>
    <P>
        text1
        <SPAN STYLE="color:red;">
            text2_in_rot;
        </SPAN>
        text3_wie_text1
    </P>
</BODY>

```

4.3.2.1.4. window.event Objekt des Netscape

Instanz aller Ereignisse im Fenster

Ein Ereignis ist ein browserinternes Signal zu einer getätigten Aktion anhand des HTML-Elementes. Das Signal zeigt an, dass eine Aktion erfolgt ist bzw. noch andauert. Typisches Ereignis ist das Mausklicken des Users auf ein HTML-Element: Klickt der User, so wird die Aktion "Klick" per Ereignis "onclick" signalisiert, wenn das HTML-Element klickbar ist, also die Aktion "Klick" unterstützt.

Ein Ereignis kann von einem konkreten HTML-Element bzw. Objekt nur dann ausgelöst werden, wenn es Events überhaupt unterstützt. Nicht alle HTML-Elemente unterstützen Events. Die Event-Unterstützung von HTML-Elementen ist in der Scriptmaschine spezifisch zu jedem HTML-Element genau vordefiniert. Aus dieser Menge der vordefinierten Events zum HTML-Element kann der Programmierer schöpfen und Aktionen des Users zum HTML-Element zulassen oder unterbinden. Dabei gilt die Regel: Was nicht explizit zugelassen wurde, gilt als unterdrückt, falls es keine standardmäßige Zulassung gibt.

Pro Aktionsart existiert eine Eventart. Erfreulicherweise gibt es diverse Aktionen, die von vielen HTML-Elemente unterstützt werden, vorallem eben o.g. Useraktionen. Aber es gibt diverse Aktionen, von denen der User nichts mitbekommt. Die Eventbehandlung dient also nicht ausschliesslich der Aktionsfreudigkeit des Users und Programmierers, sondern ist ein Nebeneffekt in Form der interaktiven Webseite.

Viele Objekte können erst kommunizieren, wenn sie Events erzeugen bzw. ein Signal als Voraussetzung für eine Aktion bekommen. Dabei ist das Wort "Aktion" auch als "Verhalten" zu verstehen., also als Komponente zur Steuerung von HTML-Elementen während der Anzeige des HTML-Dokumentes im Browserfenster. Typisches Event ist das Signal "onload", das ausgelöst wird, wenn das HTML-Dokument **komplett geparst** und falls nötig in das Browserfenster geladen **wurde**. Es gibt diverse Aktionen, die erst **nach** Auslösung von onload zulässig sind. Diese Aktionen betreffen auch die skriptgesteuerte Verwaltung des HTML-Dokumentes zu dessen Laufzeit.



Jedes instanzierte Objekt, das Events unterstützt, nutzt das Eventobjekt, mit dem die Verarbeitung aller unterstützten Events möglich ist (verschiedene Eventarten).

Aufgrund der Verschachtelung von HTML-Elementen müssen Events auch innerhalb der Hierarchie der HTML-Elemente **und** in der Vererbungsfolge verfügbar sein. Events können also durchgereicht werden. In Script wird üblicherweise die Punktnotation für Hierarchieebenen verwendet. Events können nur dann durchgereicht werden, wenn **beide betroffene Ebenen** diese Events unterstützen. Natürlich müssen die verschachtelten HTML-Elemente, welche Events durchreichen (und diese eventuell zuvor auswerten, wobei das Kind anders auf das Event reagieren kann als Eltern, die aber über das Ereignis immer informiert werden **sollten**), das Entstehen von Ereignissen auch überwachen, denn es muss nicht bekannt sein, wann und wo das Ereignis entsteht. Diese Überwachung kann in Verbindung mit Script genau gesteuert werden.

Es existiert pro HTML-Element bzw. Objekt, das Events unterstützt, eine je nach Eventart vordefinierte Standardbehandlung. Aber Events können abweichend von der Standardbehandlung durch private Eventhandler in Script verarbeitet werden. Auch die Eventüberwachung kann in Script genau gesteuert werden: Es muss also nicht sein, dass ein Kind ein Event seinen Eltern mitteilt (darüber entscheidet der Programmierer).

Aufgrund dieser Tatsache sind Script-Methoden der Eventverwaltung nicht im Objekt event implementiert, sondern im Objekt, das das Event erzeugt. Die Implementierung ist objekt-spezifisch aber genormt. Es gibt also nicht viele jedoch wie immer browserspezifische Methoden.

Instanz-Erzeugung in HTML:

Beispiele zur Kodierung von onXXX="..." im HTML-Tag des Objektes, wobei onXXX ein im Objekt implementiertes Event ist, z.B. onclick.

Übliche Kodierung:

```
<HEAD>
  <SCRIPT>
    function InputButton_Event_onclick_Handler()
    {alert("onclick erkannt");}
  </SCRIPT>
</HEAD>
<BODY>
  <INPUT TYPE=button
    NAME="ID_Button"
    VALUE="Test"
    onclick="InputButton_Event_onclick_Handler()"
  >
</BODY>
```

Alternative 1:

```
<HEAD>
  <SCRIPT>
    function InputButton_Event_onclick_Handler()
    {alert("onclick erkannt");}
  </SCRIPT>
</HEAD>
<BODY>
  <INPUT TYPE=button
    NAME="ID_Button"
    VALUE="Test"
  >
  <SCRIPT>
    // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
    // eine im HEAD deklarierte Funktion

    // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
    // geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
    ID_Button.onclick = InputButton_Event_onclick_Handler; //ohne ()
  </SCRIPT>
</BODY>
```

Alternative 2:

```
<BODY>
  <INPUT TYPE=button
    NAME="ID_Button"
    VALUE="Test"
  >
  <SCRIPT>
    // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
    // eine im BODY deklarierte Funktion

    // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
```



```
//      geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
ID_Button.onclick = InputButton_Event_onclick_Handler; //ohne ()

function ID_Button.onclick()
{alert("onclick erkannt");}

</SCRIPT>
</BODY>
```

Alternativ 3:

```
<HEAD>
<SCRIPT>
    function InputButton_Event_onclick_Handler()
    {alert("onclick erkannt");}
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE=button
        NAME="ID_Button"
        VALUE="Test"
    >
    <SCRIPT>
        // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
        //      eine im BODY deklarierte Funktion, die eine Funktion im HEAD aktiviert

        // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
        //      geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
        function ID_Button.onclick()
        {InputButton_Event_onclick_Handler();}
    </SCRIPT>
</BODY>
```

Events bei sich überlagernden Objekten:

Beispiel: Sollte ein Image mit seiner Erzeugung ein anderes Image überlagern, so ist die Eventübergabe von und an das untere Image unterbrochen: Ereignisse onXXX kommen nicht mehr durch, solange das untere Bild nicht den **Fokus** erhält. Alternativ ist die Style-Eigenschaft z-index zu kodieren und dann der z-index auf den obersten zu setzen.

4.3.2.1.4.1. Eventarten (Auswahl)

onabort	Ladevorgang eines Bildes bricht ab z.B. wegen Useraktion Stop Tag Objekt image NS ab 3.x
onblur	Ereignis, das direkt vor dem De-Fokussieren ausgelöst wird Tag z.B. <FRAME>, <INPUT>, <TEXTAREA> NS ab 3.x
onchange	Ereignis ausgelöst, wenn Inhalt von Eingabefeld Textarea Auswahlliste Radiobox Checkbox durch User verändert wurde und nur bei Eingabefeld, Textarea, Auswahlliste diese de-fokussiert wurden (bei Radiobox und Checkbox sofort Ereignisauslösung) Handler hat innerhalb <OPTION> eines <SELECT> keine Wirkung, also nur innerhalb <SELECT> kodieren Tag z.B. <INPUT>, <SELECT>, <TEXTAREA> NS ab 3.x
onclick	Ereignis ausgelöst, wenn Maus das Objekt angeklickt hat: 1 Klick = Maustaste drücken und loslassen im Handler return false; kodieren für Links, Formularelemente, wenn eine Aktion nicht erwünscht ist Bsp.: bei Link wird dieser nicht ausgeführt, obwohl angeklickt Tag z.B. <A>, <BODY>, <INPUT> Objekt document, button, submit, reset, checkbox NS ab 3.x
ondblclick	Ereignis ausgelöst, wenn Doppelklick auf Objekt durch linke bzw. rechte Maustaste erfolgt 1 Klick = Maustaste drücken und loslassen Tag fast alle Objekt link NS ab 4.x



ondragdrop	<p>NS ermöglicht Drag&Drop per Maus von Dateien und Verknüpfungen auch über Fenster</p> <p>Ablauf: Klick auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen</p> <p>Ereignis ausgelöst, wenn eine Datei oder Verknüpfung verschoben wurde</p> <p>Tag <BODY></p> <p>Objekt window</p> <p>nur NS ab 4.x</p>
onerror	<p>Ereignis ausgelöst, wenn</p> <ul style="list-style-type: none"> während des Ladens eines HTML-Dokumentes ein Syntaxfehler auftritt eines Bildes ein Fehler auftritt während der Abarbeitung von Scripten ein Runtime-Error auftritt <p>sämtliche Fehlermeldungen unterbinden per</p> <pre>var RetteOnErrorHandler = window.onerror; window.onerror = null;</pre> <p>Eventhandler muss wie folgt kodiert werden:</p> <pre>function freier_name_fuer_onerror_behandlung (error_erklaerung_string, url_des_html_dokumentes_als_string, zeilen_nr_des_errors_im_html_dokument) { return true; // nur wenn true geliefert, dann // wird die browsereigene // onerror-Behandlung // unterdrückt }</pre> <p>pro Fehler ein Aufruf des Eventhandlers --> Folge von Fehlern, also Folge von Aufrufen</p> <p>Tag z.B. </p> <p>Objekt window, img</p> <p>NS ab 3.x</p>
onfocus	<p>Ereignis ausgelöst, wenn HTML-Element fokussiert wird, also das aktuelle wird</p> <p>Tag z.B. <BODY>, <DIV>, <INPUT>, <FRAMESET>, <TEXTAREA></p> <p>Objekt window</p> <p>NS ab 3.x</p>
onkeydown	<p>Ereignis ausgelöst für alle Tastenarten</p> <p>füllt nur beim ersten Tastendruck einmalig den ASCII-Code der Taste nach Eigenschaft .which (auch bei Dauerdruck der Taste nur beim ersten Tastendruck geliefert)</p> <p>Hinweis: nach String umwandeln per .fromCharCode()</p> <p>Besonderheit bei Kombination von Steuertaste mit anderer Taste (z.B. Shift-Taste und B-Taste zugleich für Großbuchstabe B):</p> <p>Ereignis wird pro Tastendruck aufgerufen, also</p> <ol style="list-style-type: none"> 1. Mal für Shift-Taste 2. Mal für B-Taste <p>keydown-Ereignis ruft automatisch das keypress-Ereignis auf, es sei denn, der keydown-Handler liefert return false;</p> <p>Tag z.B. <A>, , <TEXTAREA></p> <p>Objekt document</p> <p>NS ab 4.x</p>
onkeypress	<p>wird unmittelbar nach dem keydown-Ereignis ausgelöst allerdings nur für</p> <p>Tastenarten</p> <pre>! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ " ' ` ~ 0 bis 9 a bis z Enter Leertaste</pre> <p>Gross und Klein wird unterschieden !</p> <p>dient dem Erkennen von Tastaturkombinationen, die nicht vom Ereignis keydown abgefangen werden</p> <p>der Aufruf des keypress-Handlers ist nur möglich, wenn der keydown-Handler return true; liefert</p> <p>Tag z.B. <A>, , <TEXTAREA></p> <p>Objekt document</p> <p>NS ab 4.x</p>



onkeyup	<p>Ereignis ausgelöst, wenn gedrückte Tastatur-Taste jeder Art losgelassen wird wird automatisch nach Ereignis keypress ausgelöst besitzt vordefiniertes Schlüsselwort KEYUP für captureEvents also document.captureEvent(Event.KEYUP);</p> <p>Tag z.B. <A>, , <TEXTAREA></p> <p>Objekt document</p> <p>NS ab 4.x</p>
onload	<p>Ereignis ausgelöst, wenn vollständig geladen wurde HTML-Dokument mit all seinen Elementen jeder Art (BODY-Teil des Dokumentes wurde komplett geparkt)</p> <p>Framset (innerhalb <FRAMESET> kodieren)</p> <p>Bild aber bei animiertem Bild (Gif-Datei): Ereignis load bei jedem Bildwechsel der Animation ausgelöst Ereignis muss also für jedes Teil-Bild behandelt werden</p> <p>DIV Layer (nicht ab NS-Version 6.x)</p> <p>Tag z.B. <BODY>, <DIV>, <FRAMESET>, </p> <p>Objekt window, img</p> <p>NS ab 3.x</p>
onmousedown	<p>Ereignis ausgelöst mit drücken der linken oder rechten Maustaste (mittlere nicht !) auch in Verbindung mit Umschalt (Shift) und Alt</p> <p>siehe Event-Eigenschaft .which return false; unterbindet immer die die Ausführung des Eventhandlers return false bei rechter Maustaste auf das Object document: Es wird das Kontextmenü gesperrt.</p> <p>Tag z.B. <A>, <DIV>, <INPUT TYPE="button"></p> <p>Objekt document</p> <p>NS ab 4.x</p>
onmousemove	<p>Ereignis bei jeder Mausbewegung ausgelöst Auswertung für den Bereich eines Objektes durch dessen Eventhandler</p> <p>Tag z.B. <A>, <DIV>, </p> <p>Objekt document</p> <p>NS ab 4.x</p>
onmouseout	<p>Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt</p> <p>Tag z.B. <A>, <DIV>, </p> <p>Objekt document</p> <p>NS ab 3.x</p>
onmouseover	<p>Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt</p> <p>Tag z.B. <A>, <DIV>, </p> <p>Objekt document</p> <p>NS ab 3.x</p>
onmouseup	<p>Ereignis ausgelöst mit Loslassen der linken Maustaste (rechte und mittlere nicht !) siehe Event-Eigenschaft .which return false; unterbindet immer die die Ausführung des Eventhandlers</p> <p>Tag z.B. <A>, <DIV>, <INPUT TYPE="button"></p> <p>Objekt document</p> <p>NS ab 4.x</p>
onmove	<p>Ereignis ausgelöst, solange ein Fenster verschoben wird</p> <p>Tag keine</p> <p>Objekt window</p> <p>NS ab 4.x</p>
onreset	<p>Ereignis VOR dem Rücksetzen eines Formulars ausgelöst (Rücksetzen per Reset-Button bzw. Neuladen des Dokumentes) dient zum Abfangen der Resetaktion des Users und eventueller Unterbindung des Resets z.B. wegen falscher Ausfüllung des Formulars</p> <p>Reset wird nicht ausgelöst, wenn Handler return false; liefert</p> <p>Tag <FORM></p> <p>Objekt form</p> <p>NS ab 3.x</p>
onresize	<p>Ereignis mit ausgelöst bei Größenänderung des Fensters per Maus: Maus auf Fensterrahmen</p>



es erscheint das <> Symbol
linke Maustaste drücken, gedrückt lassen und ziehen
linke Maus loslassen dann wird Ereignis ausgelöst

Tag keins
Objekt window
NS ab 3.x

onselect Ereignis ausgelöst wenn sich Textmarkierung ändert
Tag z.B. <INPUT TYPE="text">, <TEXTAREA>
NS ab 3.x

onsubmit Ereignis VOR dem Abschicken eines Formulars ausgelöst
(Abschicken per Submit-Button)
dient zum Abfangen der Resetaktion des Users und eventueller Unterbindung
des Submits z.B. wegen falscher Ausfüllung des Formulars
Submit wird nicht ausgelöst, wenn Handler return false; liefert
Tag <FORM>
Objekt form
NS ab 3.x

onunload Ereignis ausgelöst, wenn das HTML-Dokument verlassen wird durch
Schliessen Browserfenster
Wechsel zu anderer Seite auch per Browser-Button
window.document.open()
Tag z.B. <BODY>, <FRAMESET>
Objekt window
NS ab 3.x

4.3.2.1.4.2. **Eigenschaften (Auswahl)**

.data Zeiger auf Feld der per Urls aller per Drag & Drop auf das HTML-Dokument abgelegten Objekte
NS ab 4.x mit signiertem Script

.height Höhe in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde
NS ab 4.x

.layerX Ereignis ist nicht resize:
horizontale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich
dessen linker oberer Ecke (0,0)
Ereignis ist resize:
neue Breite in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Breite verändert
wurde
NS ab 4.x

.layerY Ereignis ist nicht resize:
vertikale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich
dessen
linker oberer Ecke (0,0)
Ereignis ist resize:
neue Höhe in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Höhe verändert
wurde
NS ab 4.x

.modifiers Bitmaske für Zustand der Steuertasten Shift, Alt und Strg
dient zur Ermittlung der Steuertaste per vordefinierter Maske durch bitweise UND-Verknüpfung mit der
Bitmaske: Ergibt die Verknüpfung 1, also true, so ist Steuertaste gedrückt worden
vordefinierte Maske für
Alt-Taste "Event.ALT_MASK"
Strg-Taste "Event.CTRL_MASK"
Shift-Taste "Event.SHIFT_MASK"
Bsp: return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0);
// nicht logisches UND (&) sondern bitweises UND also &
NS ab 4.x

.pageX horizontale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
NS ab 4.x

.pageY vertikale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
NS ab 4.x

.screenX horizontale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms
NS ab 4.x



.screenY	vertikale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms NS ab 4.x
.target	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per onXXX = "" oder Bezug auf den logischen Namen des HTML-Elementes if (logischer_name = Ereignis.target) NS ab 4.x
.type	enthält die Event-Art z.B. abort NS ab 4.x
.which	enthält bei gedrückter Maustaste: 1 für linke Taste 2 für mittlere Taste 3 für rechte Taste Tastatur ASCII-Code der Taste Hinweis: Umwandlung in String per String.fromCharCode(TastenKodeASCII) NS ab 4.x
.width	Breite in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde NS ab 4.x
.x	identisch mit .layerX NS ab 4.x
.y	identisch mit .layerY NS ab 4.x

4.3.2.1.4.3. Methoden

keine

Events können zusätzlich nur durch Methoden anderer Objekte verwaltet werden.

4.3.2.1.4.4. Prinzipien der Eventbehandlung des Netscape

Die Standard-Eventbehandlung wird beim Netscape immer vom window-Objekt selbst realisiert. Damit landen alle Events immer beim obersten Objekt. Eine davon abweichende Eventbehandlung ist zu programmieren.

4.3.2.1.4.4.1. Event des Netscape einer Nicht-Standardbehandlung unterziehen

4.3.2.1.4.4.1.1. Event und Eventhandler dem Objekt window zuordnen (captureEvents(event_liste))

Diese Methode unterbindet die Ereignisregistrierung des Browsers. Damit MUSS das Ereignis von einem Eventhandler verarbeitet werden. Der Eventhandler kann mehrere Ereignisse verarbeiten.

Achtung: In Eventhandlern, die Mausereignisse oder Tastaturereignisse verwalten, sollte kein alert() etc. programmiert werden, da alert() selbst solche Events erzeugt und damit die eigentlichen Events aufhebt. Alternative: Meldungen in der Statuszeile des Fensters anzeigen.

Start der Registrierung des Events durch:

```
window.captureEvents(Event.event_schluesselwort_liste);
```

event_schluesselwort_liste: Folge von Eventbezeichnern mit Trennung durch | also logisches Oder

Bsp. für Eventbezeichner

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS
onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER
onkeydown	Event.KEYDOWN
onkeypress	Event.KEYPRESS
onkeyup	Event.KEYUP
onresize	Event.RESIZE

Beispiel:

```
function MouseDownHandler(Ereignis)
```



```

{..}

window.captureEvents(Event.MOUSEDOWN);

// Standardeventhandler überschreiben
window.onmouseover=MouseDownHandler;      // Achtung: nicht () kodieren !!

Empfehlung: Vor dem Überschreiben des Standardeventhandler diesen retten
var RetteHandler = window.onmouseover;
window.onmouseover=MouseDownHandler;

```

Wenn das Ereignis nur einmal bearbeitet und danach wieder der Standardbehandlung zugeordnet werden soll, so muss der Eventhandler vor dem return die Methode `.releaseEvents()` zum Ereignis aufrufen. Falls der Standardeventhandler gerettet wurde, dann diesen auch rückspeichern.

Wenn das Ereignis **nicht** durch nachgelagerte Eventhandler bzw. Aktionen durch deren automatischen Aufruf bearbeitet werden soll, so muss die Funktion mit `return false;` enden (sonst `return true;`;) z.B. siehe Event onreset etc. für Formular.

4.3.2.1.4.4.1.2. **Event entlang der Eventhierarchie weiterreichen**

Es besteht die Möglichkeit, dass ein privater Eventhandler für die Weitergabe des Events die Standardhierarchie benutzt oder einen ausgewählten Eventhandler aufruft.

4.3.2.1.4.4.1.2.1. **Event entlang der Nicht-Standard- Eventhierarchie weiterreichen (`handleEvent(event_objekt)`)**

Diese Methode ruft den aktuell dem Ereignis laut Parameter `event_objekt` zugeordneten Eventhandler auf. Diese Methode wird also innerhalb des Programmcodes eines Eventhandlers verwendet, der das Ereignis von einem anderen Eventhandler als Argument bekommen hat.

4.3.2.1.4.4.1.2.2. **Event entlang der Standard-Eventhierarchie weiterreichen (`routeEvent(ereignis_objekt)`)**

Diese Methode aktiviert das Weiterreichen des Events zum in der Eventhierarchie untergeordneten Eventhandler.

4.3.2.1.4.4.2. **Event des Netscape von Nicht-Standardbehandlung wieder der Standardbehandlung unterziehen**

4.3.2.1.4.4.2.1. **Standard-Eventhandler dem Objekt window zuordnen (`releaseEvents(event_liste)`)**

Diese Methode aktiviert die Ereignisregistrierung durch den Browser, ist also der Gegensatz zu `captureEvents()`. Es dürfen nur Ereignisse releast werden, die auch gecaptured wurden.

Ende der Registrierung eines Events:

```

window.releaseEvents(Event.event_schluesselwort_liste);

event_schluesselwort_liste: Folge von Eventbezeichnern mit Trennung durch | also logisches Oder

```

Bsp. für Eventbezeichner

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS
onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER
onkeydown	Event.KEYDOWN
onkeypress	Event.KEYPRESS
onkeyup	Event.KEYUP
onresize	Event.RESIZE

Beispiel:

```

function EinmaligerMouseDownHandler()
{
    .....
    window.releaseEvents(Event.MOUSEDOWN);
}

window.captureEvents(Event.MOUSEDOWN);

window.onmouseover=EinmaligerMouseDownHandler;      // Achtung: nicht () kodieren !!

```

Wenn das Ereignis **nicht** durch nachgelagerte Eventhandler bzw. Aktionen durch deren automatischen Aufruf bearbeitet werden soll, so muss die Funktion mit `return false;` enden (sonst `return true;`;) z.B. siehe Event onreset etc. für Formular.

4.3.2.1.4.4.2.2. **Event entlang der Standard-Eventhierarchie weiterreichen (`routeEvent(ereignis_objekt)`)**

Diese Methode aktiviert das Weiterreichen des Events zum in der Eventhierarchie untergeordneten Eventhandler.



4.3.2.1.4.4.3. Eventbehandlung durch eine Fremdseite mit signiertem Script

Eventbehandlung des Netscape für eine Seite mit Frame, in dem ein fremdes HTML-Dokument angezeigt wird:

Es ist möglich, dass die fremde Seite die Ereigniskontrolle derjenigen Seite übernimmt, die das Frame enthält, in dem die Fremdseite angezeigt wird. Dieser Trojanereffekt ist nur dann realisierbar, wenn die Fremdseite ein signiertes Script enthält: Dieses Script muss sich beim Eigentümer der Seite, die den Frame besitzt, das Recht (Privileg) zur Übernahme der Ereigniskontrolle beschaffen. Dazu wird der User, in dessen Browser die Seite mit dem Frame läuft, befragt. Der Netscape hat einen Privileg-Manger integriert.

4.3.2.1.4.4.3.1. Beschaffung der Rechte "UniversalBrowserWrite" bzw. "UniversalBrowserWrite" per Privilegmanger

Die fremde Seite kann z.B. folgenden Code besitzen:

```
netscape.security.PrivilegeManager.enablePrivilege("UniversalBrowserWrite");
window.enableExternalCapture();
window.captureEvents(Event.CLICK | Event.MOUSEDOWN);
```

Bei Ausführung des Codes wird der Manager aktiv, um die Rechtegenehmigung durch den User zu beschaffen, da dieser Code (Script) vom User signiert sein muss.

Der Netscape-Privilegmanger sichert ab, dass der Browseruser gefragt wird, ob er eine Privilegänderung gegenüber den Standard-Privilegien dulden will.

Für ein signiertes Script muss das Privileg "UniversalBrowserWrite" bzw. "UniversalBrowserWrite" vom Privilegmanger angefordert sein.

4.3.2.1.4.4.3.2. Eventbehandlung durch Fremde Seite aktivieren (enableExternalCapture())

Diese Methode aktiviert die Kontrolle einer Fremdseite, die in einem Frame dargestellt wird, auf die Seite, die den Frame inne hat. Die Ereignissüberwachung in einem Fenster wird somit aktiv.

Diese Methode benötigt ein signiertes Script.

Das Event muss mit captureEvents() dem Window-Objekt zugeordnet werden.

Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !

4.3.2.1.4.4.3.3. Eventbehandlung durch Fremde Seite deaktivieren (disableExternalCapture())

Diese Methode deaktiviert die Kontrolle einer Fremdseite, die in einem Frame dargestellt wird, auf die Seite, die den Frame inne hat. Die Ereignissüberwachung in einem Fenster wird somit aktiv.

Diese Methode benötigt ein signiertes Script.

Das Event muss mit captureEvents() dem Window-Objekt zugeordnet werden.

Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !

4.3.2.1.4.4.4. Beispiel

Es ist zu beachten, dass beim Netscape das Ereignis nicht als Objekt direkt angesprochen werden darf, sondern immer nur über den Parameter zum Eventhandler. Der Bezeichner des Paramaters ist frei wählbar. NS weiss, dass dieser Parameter das Eventobjekt referenziert. Es ist also auch der Bezeichner "event" wählbar.

Beim Internet Explorer muss das Ereignis im Eventhandler immer direkt über das Objekt event (Unterojekt des Objektes window) behandelt werden. Daher ist für den Eventhandler kein Parameter nötig.

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript">
<!--
    function MouseDownFuerInputButton(Ereignis)
    { ... }

    function OnMouseDownEventWeiterreichen(Ereignis)
    {window.routeEvent(Ereignis);} // In der Hierarchie weiterreichen

    window.captureEvents(Event.MOUSEDOWN);

    window.onmousedown= OnMouseDownEventWeiterreichen;
                          // Achtung: ohne () kodieren!!

// -->
</SCRIPT>
</HEAD>
<BODY>
    <FORM>
        <INPUT TYPE=button
              VALUE="Weiterreichen"
              onmousedown= "MouseDownFuerInputButton();"
              >
```



```

        </FORM>
    </BODY>
</HTML>

```

Ablauf: Head-Teil: Der Script-Teil im Head wird zuerst abgearbeitet.
 Body-Teil: Es wird auf das Input-Button gedrückt.
 Wegen captureEvents() wird das Mausereignis ZUERST von
 OnMouseDownEventWeiterreichen
 bearbeitet. Damit wird auf das Weiterleiten aktiviert. Das untergeordnete Element ist der Input-
 Button. Damit wird MouseDownFuerInputButton aktiviert.

Hinweis: Dieses Beispiel hinkt, da die Standardbehandlung letztendlich das gleiche bewirkt aber auf anderen Wegen.

4.3.2.1.4.5. **Beispiele zur Eventbehandlung des Netscape**

4.3.2.1.4.5.1. **Formular**

Es wird das Click-Event abgefangen und mit einer Funktion bearbeitet.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE ="Javascript">
<!--
    // Browser feststellen
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    if (ns)
    {
        // nur für NS: Eventart CLICK wird abgefangen
        window.captureEvents(Event.ONCLICK);
    }

    // soll das Fenster selbst den Click-Eventhandler 1 bekommen, so hier kodieren
    // window.onclick=click_auswertung_1;
    // WICHTIG ist, dass das Fenster auch das Event WEITERLEITET !! nach unten

    function click_auswertung_1(Ereignis)
    {
        // folgende alternative Varianten sind möglich

        // nur für NS:
        // in der Objekt-Hierarchie weiterleiten
        if (ns)
        {routeEvent(Ereignis);}

        // für NS und IE:
        // nichts tun
        return true;

        // im aktuellen Fenster und dessen Dokument das Formular anwählen
        // und dem Eventhandler des 2. Buttons das Ereignis übergeben,
        // wobei der Eventhandler des 2. Buttons dieses Ereignis verarbeiten muss
        self.document.logischer_form_name.logischer_button_name_2.handleEvent(Ereignis);

        // die Auswertung hier kodieren --> bitte kein alert(); !!!
        if (ie)
        {
            Ereignis=event.button;
            // hier das Maus-Ereignis allgemein erfasst
            //          0      keine Maustaste gedrückt
            //          1      linke Maustaste gedrückt
            //          2      rechte Maustaste gedrückt
            //          4      mittlere Maustaste gedrückt
            //          Kombination aus 1 bis 4 für Maustastenkombination
            //          z.B. 3 = linke UND rechte Maustaste gedrückt (1 + 2 = 3)
            //          7 = alle Maustasten gedrückt (1 + 2 + 3 + 4 = 7)
            //          nur Internet Explorer ab 4.x
            .....
        }
        else
        {

```



```

        if (ns)
        {
            .....
        }
    }

    function click_auswertung_2(Ereignis)
    { // analog zu click_auswertung_1 }

// -->
</SCRIPT>
</HEAD>
<BODY ....>
    <FORM NAME="logischer_form_name">
        <INPUT          TYPE=button
                        NAME="logischer_button_name_1"
                        onclick="click_auswertung_1();"
        >
        <INPUT          TYPE=button
                        NAME="logischer_button_name_2"
                        onclick="click_auswertung_2();"
        >
    </FORM>
</BODY>
</HTML>

```

4.3.2.1.4.5.2. Tastatur-Eventbehandlung beim Netscape

Der Event-Handler muss als Parameter das gewünschte Event übergeben bekommen

4.3.2.1.4.5.2.1. Tastatur-Eventarten

onkeydown Ereignis ausgelöst für alle Tastenarten der Tastatur
 liefert einmalig den ASCII-Code der Taste nach Eigenschaft .which, wenn Taste das erste Mal gedrückt wird (also auch einmalig bei Dauertastendruck)
 Hinweis: nach String umwandeln per var string_name=fromCharCode(..)
 Besonderheit bei Kombination Steuertaste und andere Taste
 z.B. Shift-Taste und B-Taste zugleich für Großbuchstabe B
 Ereignis wird pro Tastendruck aufgerufen, also
 1. Mal für Shift-Taste
 2. Mal für B-Taste

keydown-Ereignis ruft automatisch das keypress-Ereignis auf, wenn keydown-Handler return false; liefert
 .which ASCII-Code der gedrückten Taste
 wenn 0 so keine Taste gedrückt
 wobei bei Buchstaben unterschieden wird
 Umwandlung in String per String.fromCharCode(TastencodeASCII)

Zusatzanalyse möglich per Event-Eigenschaft .modifiers:
 Bitmaske für Zustand der Steuertasten Shift, Alt und Strg
 dient zur Ermittlung der Steuertaste per vordefinierter Maske
 durch bitweise UND-Verknüpfung mit der Bitmaske:
 ergibt die Verknüpfung 1, also true, so ist Steuertaste gedrückt worden

vordefinierte Maske für

Alt-Taste	"Event.ALT_MASK"
Strg-Taste	"Event.CTRL_MASK"
Shift-Taste	"Event.SHIFT_MASK"

Unterbindung des Aufrufes des keypress-Handlers:
 keydown-Handler muss return false; bzw. event.returnValue=false; liefern

onkeypress wird unmittelbar nach dem keydown-Ereignis ausgelöst allerdings nur für Tastenarten
 ! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ | ' " ` ~
 0 bis 9
 a bis z mit Unterscheidung Gross oder Klein
 Enter
 Leertaste

und dient dem Erkennen von Tastaturkombinationen, die nicht vom Ereignis keydown abgefangen werden
 Aufruf des keypress-Handlers ist **nur** möglich, wenn der keydown-Handler return true; liefert

onkeyup Ereignis ausgelöst, wenn gedrückte Tastatur-Taste jeder Art losgelassen wird
 wird automatisch nach Ereignis keypress ausgelöst
 besitzt vordefiniertes Schlüsselwort KEYUP für captureEvents
 also document.captureEvent(Event.KEYUP);

4.3.2.1.4.5.2.2. Tastatur-Eigenschaft

.modifiers Bitmaske für Zustand der Steuertasten Shift, Alt und Strg
 dient zur Ermittlung der Steuertaste per vordefinierter Maskedurch bitweise UND-Verknüpfung mit der
 Bitmaske: ergibt die Verknüpfung 1, also true, so wurde die Steuertaste gedrückt



vordefinierte Maske für
 Alt-Taste "Event.ALT_MASK"
 Strg-Taste "Event.CTRL_MASK"
 Shift-Taste "Event.SHIFT_MASK"

Beispiel:

```
return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0); // nicht logisches UND && sondern bitweises UND &
```

.which enthält ASCII-Code der Tastatur-Taste
 Umwandlung in String per String.fromCharCode(TastenKodeASCII)

.type enthält die Event-Art z.B. abort

.target Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde
 entweder im HTML-Tag definieren per onXXX = ""
 oder Bezug auf den logischen Namen des HTML-Elementes
 if (logischer_name = Ereignis.target)

4.3.2.1.4.5.2.3. Beispiel zur Tastatur-Eventbehandlung

```
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
```

```
var ns = document.layers ? true : false;
var ie = document.all ? true : false;
```

```
// Funktionen für Tasten aller Art holen
function NS_IE_TastenKodeHolen_ASCII(Ereignis)
{
    if (ie)
    {return event.keyCode;}
    else
    {
        if (ns)
        {return Ereignis.which;}
    }
}
```

```
function NS_IE_TastenKodeHolen_String(TastenKodeASCII)
{
    if (TastenKodeASCII == 0)
    {return "";}
    else
    {return String.fromCharCode(TastenKodeASCII);}
}
```

```
function IE_TasteDauerdruck_StatusHolen()
{return event.repeat;} // true, so Taste im Dauerdruck, sonst false
```

```
// Funktionen für Shift-Taste
// liefern true für Shift-Taste gedrückt; sonst false
```

```
function NS_IE_ShiftTaste_StatusHolen(Ereignis)
{
    if (ie)
    {return event.shiftKey;}
    else
    {
        if (ns)
        {return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0); }
    }
}
```

```
function IE_ShiftLeftTaste_StatusHolen()
{return event.shiftLeft;}

function IE_ShiftRightTaste_StatusHolen()
{return (event.shiftKey AND !event.shiftLeft);}
```

```
// ***** Funktionen für Alt-Taste analog zu Shift aber für NS mit Event.ALT_MASK
```

```
// ***** Funktionen für Strg-Taste analog zu Shift aber für NS mit Event.CTRL_MASK
```

```
// -->
</SCRIPT>
```



4.3.2.1.4.5.3. Mouse-Eventbehandlung beim Netscape

Der Event-Handler muss als Parameter das gewünschte Event übergeben bekommen.

4.3.2.1.4.5.3.1. Mouse-Eventarten

onclick	Ereignis ausgelöst, wenn Maus das Objekt angeklickt hat: Taste drücken und loslassen im Handler return false; kodieren für Links, Formularelemente, wenn eine Aktion nicht erwünscht ist
ondblclick	Ereignis ausgelöst, wenn durch linke bzw. rechte Maustaste ein Doppelklick auf Objekt erfolgt 1 Klick = Maustaste drücken und loslassen
onlosecapture	Ereignis ausgelöst, wenn releaseCapture() für mousemove, mouseover und mouseout ausgeführt wurde der User mit der Maus das Browserfenster verlässt
onmousedown	Ereignis ausgelöst mit Drücken der linken oder rechten Maustaste (mittlere nicht !) auch in Verbindung mit Umschalt (Shift) und Alt siehe Event-Eigenschaft .which return false; unterbindet immer die die Ausführung des Eventhandlers return false bei rechter Maustaste auf Object document wird das Kontextmenü gesperrt
onmousemove	Ereignis bei jeder Mausbewegung ausgelöst Auswertung für den Bereich eines Objektes durch dessen Eventhandler
onmouseout	Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt
onmouseover	Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt
onmouseup	Ereignis ausgelöst mit Loslassen der linken Maustaste (rechte und mittlere nicht !) siehe Event-Eigenschaft .which return false; unterbindet immer die die Ausführung des Eventhandlers

4.3.2.1.4.5.3.2. Mouse-Event-Eigenschaften

.height	Höhe in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde
.layerX	Ereignis ist nicht resize: horizontale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis ist resize: neue Breite in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Breite verändert wurde
.layerY	Ereignis ist nicht resize: vertikale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis ist resize: neue Höhe in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Höhe verändert wurde
.pageX	horizontale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
.pageY	vertikale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
.screenX	horizontale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirmes
.screenY	vertikale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirmes
.target	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per onXXX = "" oder Bezug auf den logischen Namen des HTML-Elementes if (logischer_name = event.scrElement)
.type	enthält die Event-Art z.B. abort
.which	enthält bei gedrückter Maustaste: 1 für linke Taste 2 für mittlere Taste 3 für rechte Taste
.width	Breite in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde
.x	identisch mit layerX
.y	identisch mit layerY

4.3.2.1.4.5.4. Lade-Ereignisse beim Netscape

4.3.2.1.4.5.4.1. Lade-Ereignis für Bild

onabort Ladevorgang eines Bildes bricht ab z.B. wegen Useraktion Stop

onload Ereignis ausgelöst, wenn Bild vollständig geladen wurde
 aber bei animiertem Bild (Gif-Datei):
 Ereignis load bei jedem Bildwechsel der Animation ausgelöst
 es kann also für jedes Bild das Ereignis behandelt werden

4.3.2.1.4.5.4.2. Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)

onload Ereignis ausgelöst, wenn vollständig geladen wurde
 HTML-Dokument mit all seinen Elementen jeder Art
 Framset (innerhalb <FRAMESET> kodieren)
 DIV
 Layer (nicht mehr ab NS 6.x)

onunload Ereignis ausgelöst, wenn das HTML-Dokument verlassen wird durch
 Schliessen Browserfenster
 Wechsel zu anderer Seite auch per Browser-Button
 window.document.open()

4.3.2.1.5. window.history Objekt des Netscape

Dieses Objekt ist eigentlich eine Collection als Sammlung der besuchten Webseiten (Verlauf).

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

window.history[index].eigenschaft oder history[index].eigenschaft
 window.history[index].methode() oder history[index].methode()

logischer_window_name.history[index].eigenschaft
 logischer_window_name.history[index].methode()

index: ab 0
 muss in [] kodiert sein

Eigenschaften (Auswahl):

.current Zeichenkette mit Url der aktuellen Seite
 mit signiertem Script

.length Anzahl der History-Einträge
 nur lesen

.next Zeichenkette mit Url der nächsten Seite (falls vorhanden)
 mit signiertem Script

.previous Zeichenkette mit Url der vorhergehenden Seite (falls vorhanden)
 mit signiertem Script

Methoden (Auswahl):

.back() vorhergehende Seite laut .previous laden

.forward() nächste Seite laut next laden

.go(position) Position der zu ladenden Seite innerhalb der History
 0 entspricht erste geladene Seite
 history.length-1 entspricht aktuelle Seite
 Bsp.: onClick="window.history.go(0)"

4.3.2.1.6. window.location Objekt des Netscape

Dieses Objekt beschreibt die Lage des HTML-Dokumentes auf dem Server sowie dessen Eigenschaften zum HTML-Dokument.

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

window.eigenschaft oder location.eigenschaft
 window.location.methode() oder location.methode()

logischer_window_name.location.eigenschaft
 logischer_window_name.location.methode()

Eigenschaften (Auswahl):

.hash entspricht #hashtext
 zum Anspringen eines Ankers
 hier den Ankernamen mit vorgesetztem # ablegen



.host	entspricht hostname:port lesen und schreiben
.hostname	entspricht nur hostname lesen und schreiben
.href	gesamter Url (kompletter Url) zum Anspringen eines Ankers hier den Ankernamen ohne vorgesetztem # ablegen
.pathname	entspricht aus url /dateiname bzw. /dateiname#hashtext bzw. /dateiname?searchtext lesen und schreiben
.port	entspricht port lesen und schreiben
.protocol	entspricht Protokoll mit Doppelpunkt z.B. "http:" lesen und schreiben
.search	entspricht ?searchtext lesen und schreiben

Methoden (Auswahl):

.reload([true])	wenn true entfällt: Dokument vom Cache auf Festplatte laden wenn true kodiert: Dokument vom Server und nicht Cache laden Achtung: Server kann selbst Cache haben und daraus das Dokument liefern
.replace(url)	aktuelle Seite mit neuer geladener Seite überschreiben sowie den HistoryEintrag zur aktuellen Seite überschreiben (keinen neuen bilden) --> BACK-Button wechselt danach nicht zur überschriebenen Seite, da diese in der History nicht mehr bekannt ist

4.3.2.1.7. window.navigator Objekt des Netscape

Container der Informationen über den Browser, Betriebssystem, regionale Einstellung des Users, CPU und Java-Maschine

Die Informationen sind z.T. browserherstellerspezifisch und können sich von Version zu Version ändern. Es ist daher zu empfehlen, bei einer Browserprüfung auch Javascript zu verwenden und browsersepezifische Funktionen aufzurufen. Deren Rückkehrcode liefert den Beweis, ob der gewünschte Browser auch benutzt wird vom User (siehe DOM). Ein typisches Tarnverhalten zeigt der Opera-Browser: Er gibt sich als Internet Explorer aus.

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

window.navigator.eigenschaft oder navigator.eigenschaft

window.navigator.methode() oder navigator.methode()

logischer_window_name.navigator.eigenschaft

logischer_window_name.navigator.methode()

Eigenschaften (Auswahl):

alle Eigenschaften sind nur lesbar

.appName	Browser-Codename z.B. "Mozilla"
.appName	Browsersname z.B. "Netscape"
.appVersion	Plattform und Browserversion Aufbau: versionsnummer (system; land) Bsp: "3.0B2 (Win16;I)" "4.1 (WinNT;I)" Haupt-Versionsnummer ermittelbar per parseFloat(navigator.appVersion) z.B. 4 des IE 4.1 Minor-Version --> siehe .appMinorVersion
.cookieEnabled	wenn true so Cookiesverwaltung aktiv wenn false so Cookies abgeschaltet
.language	Browser-Sprachversion z.B. "en" "de"
.mimeTypes	Zeiger auf Feld aller im Browser verfügbaren Mimetypen Index kann Typ als Zeichenkette sein z.B. "image/png"
.oscpu	nur NS 6.x in der Dokumentation nicht weiter behandelt
.platform	Browser-Betriebssystem z.B. "Win32" oder "Win16"
.plugins	Zeiger auf Feld aller im Browser verfügbaren Plugins
.product	nur NS 6.x



.productSub	nur NS 6.x
.securityPolicy	Sicherheitseinstellungen vom Netscape nur NS 6.x
.userAgent	Browser-Codename UND -Version Bsp: "Mozilla / 3.0B2 (Win16;I)"
.vendor	nur NS 6.x
.vendorSub	nur NS 6.x
Methoden (Auswahl):	
.javaEnabled()	liefert true, wenn Browser Java kann und Java nicht abgeschaltet ist
.preference()	Benutzereinstellungen des Browsers verändern benötigt Privileg-Manger
.savePreferences()	aktuelle Benutzereinstellungen des Browsers sichern benötigt Privileg-Manger

4.3.2.1.7.1. *window.navigator.plugins Collection des Netscape*

Feld der Zeiger aller Plugins

Feldeintrag ist true, wenn Plugin im Feld vorhanden ist, also der Browser das Plugin besitzt.

Erzeugung:

keine, da vom Browser bereitgestellt

Zugriff:

```
zeiger_auf_navigator_objekt.plugins[index].eigenschaft
zeiger_auf_navigator_objekt.plugins[index].methode()
```

index: Zeichenkette z.B. "Shockwave"
Integer Index ab 0
muss in [] kodiert sein

```
zeiger_auf_navigator_objekt    navigator
                                window.navigator
                                logischer_window_name..navigator
```

Beispiel: Auf Adobe Acrobat-Reader-Plugin prüfen

```
<BODY>
  <SCRIPT LANGUAGE="JavaScript1.2">
    <!--
      if (navigator.plugins["Adobe Acrobat"] != null)
      {
        document.write("<EMBED SRC='test.pdf' WIDTH=600 HEIGHT=800>");
        document.write("<NOEMBED> .....</NOEMBED>");
      }
      else
      {document.write("Kein Adobe-Plugin !"); }
    //-->
  </SCRIPT>
</BODY>
```

Beispiel: Alle Plugins auflisten

```
<BODY>
  <SCRIPT LANGUAGE="JavaScript1.2">
    <!--
      for (i=0; i < navigator.plugins.length; i++)
      {
        document.writeln(navigator.plugins[i].name);
        document.writeln(navigator.plugins[i].description);
        document.writeln(navigator.plugins[i].filename);
      }
    //-->
  </BODY>
```

Eigenschaften:

.description	Zeichenkette der Plugin-Kurzbeschreibung als String
.filename	Plugin-Dateiname als String
.length	Anzahl der Plugins, ab 1
.mimeType	Zeiger auf navigator.mimeType Collection Beispiel für Zeigerbezug: navigator.plugins.mimeType[index].eigenschaft
	mit index String Mimetype Integer als Index ab 0 muss in [] kodiert sein
.name	Name des Plugin als String

Methoden:



.refresh()	Funktionswert
	true HTML-Dokument mit seinem per EMBED eingebetteten Objekten neu laden falls Plugin noch nicht installiert ist, so es dabei installieren
	false kein Refresh
.taintEnabled()	Data-Tainting (Daten verwerfen)

4.3.2.1.7.2. **window.navigator.mimeTypes Collection des Netscape**

Feld aller Mimetypen also Dateitypen, die Browser für Plugin und Dateiverarbeitung verarbeiten kann bzw. soll. Als Index dient der Mimetyp als String oder ein Integerwert ab 0.

Ein Plugin wird einem Mimetyp zugeordnet.

Hinweis: Ab dem IE 6.0 werden keine Plugins mehr unterstützt. Es muss ein Active-X-Control verwendet werden

Die Eigenschaft .enablePlugin enthält den Zeiger auf das installierte Plugin (sonst null-Zeiger). Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.

Das Speichern der Zeiger **aller** Plugin-Objekte erfolgt in der Collection window.navigator.plugins.

Feldelement:

Reihenfolge der Feldelemente ist browserspezifisch.

Es besteht die Möglichkeit, dass Mimetypen, die keine Plugins repräsentieren, ebenfalls mit in der Collection liegen

Bsp.: "image/jpeg".

Beispiel für einen Mimetyp für echten Plugin: "application/pdf" für Adobe Acrobat-Plugin

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

zeiger_auf_navigator_objekt.mimeTypes[mime_typ].eigenschaft

mime_typ: Integer, ab 0
String, der als Feldindex dient z.B. "text/html"
muss in [] kodiert sein

zeiger_auf_navigator_objekt navigator
window.navigator
logischer_window_name.navigator

Beispiel 1: Feld auslesen

```
for (var Index = 0; Index < navigator.mimeTypes.length; Index++)
{ alert(    navigator.mimeTypes[i].type        + "\n"
+ navigator.mimeTypes[i].suffixes        + "\n"
+ navigator.mimeTypes[i].description
);
}
```

Beispiel 2: Beschreibung zum Plugin anzeigen

```
if (navigator.mimeTypes["application/pdf"])
{alert(navigator.mimeTypes["application/pdf"].description);}
```

Beispiel 3: Daten an Plugin für VRML-Dateiverarbeitung übergeben

```
if (navigator.mimeTypes["x-world/x-vrml"])
{
    if(navigator.mimeTypes["x-world/x-vrml"].enablePlugin != null)
    {
        document.write('<OBJECT DATA="zyeplin.wrl" WIDTH="400" HEIGHT="300"></OBJECT>');
    }
}
```

Beispiel 4: Suffix prüfen

```
if (navigator.mimeTypes["image/jpeg"])
{alert(navigator.mimeTypes["application/pdf"].suffixes);}
```

Eigenschaften:

.description	Zeichenkette mit der Kurzbeschreibung des Mimetyps
.enablePlugin	Zeiger auf ein Plugin null-Zeiger, wenn Plugin nicht installiert ist Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.
.length	Anzahl der Feldelemente, ab 1
.suffixes	Liste aller erlaubten Dateisuffixe zum Mimetyp



.type Mimetyp z.B. "text/html"

Methoden:

keine

4.3.2.1.8. **window.screen Objekt des Netscape**

Dieses Objekt beschreibt den Bildschirm

Die linke obere Browserfensterecke ist der Ursprung des Grafiksystems, also (0,0) mit (x,y)

x für horizontal

y für vertikal

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

window.screen.eigenschaft oder screen.eigenschaft

logischer_window_name.screen.eigenschaft

Eigenschaften:

.availHeight	maximal verfügbare Fensterhöhe in Pixel laut max. Bildschirmauflösung
.availLeft	minimale Pixel-X-Position der Browserfensterecke links oben (horizontal) hängt von Position der Windows-Taskleiste ab
.availTop	minimale Pixel-Y-Position der Browserfensterecke links oben (vertikal) hängt von Position der Windows-Taskleiste ab
.availWidth	maximale verfügbare Fensterbreite in Pixel laut max. Bildschirmauflösung
.colorDepth	Farbtiefe der aktuellen Farbpalette, also Anzahl der verfügbaren Farben
.height	aktuelle Höhe in Pixel der Bildschirmauflösung
.left	Abstand links in Pixel
.pixelDepth	aktuelle Anzahl Farbbits pro Bildpunkt laut aktueller Bildschirmauflösung 1 oder 4 oder 8 oder 15 oder 16 oder 24 oder 32
.top	Abstand oben in Pixel
.width	aktuelle Breite in Pixel der Bildschirmauflösung

Methoden:

keine

4.3.2.2. **window Objekt des Internet Explorer**

Objekt eines Browserfenster das im Browser erzeugt wird
das in browserspezifischen Versionen erzeugt werden kann

Das window Objekt hat diverse Zeiger auf andere Objekte:

z.B. beim IE und NS:

.document	Zeiger auf das Objekt document im Fenster
.event	Zeiger auf das Objekt event im Fenster
.history	Zeiger auf das Objekt history (Verlauf)
.location	Zeiger auf das Objekt location
.navigator	Zeiger auf das Objekt navigator

z.B. beim IE:

.screen	Zeiger auf das Objekt screen
---------	------------------------------

Diese Zeiger dienen **nur** der Zuordnung der Objekte zum Fenster. Solange es sich um das aktuelle Fenster handelt, kann also in der Punktnotation die Kodierung von window entfallen.

Das Objekt window beschreibt obige Objekt nicht: Aus Sicht des HTML-DOM sind Objekte, die im DOM hinterlegt sind, keine Kinder des Fensters, denn das ist im DOM nur **durch** das HTML-Dokument present..

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer



Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
klick ins Fenster per aktivem Popup

Der Absturz ist "read"-Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Popupfenster geblockt. Sie können den Popupblocker deaktivieren

oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster



einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popublocker des IE bemerkt aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popublocker verwaltet wird.

Der Popublocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popublocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popublockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popublockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

```
window.focus();
```

```
window.document.focus();
```

```
if(document.body!=null)
```

```
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
```

```
{ document.body.focus();}
```

```
}
```

```
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
```

```
popupzeiger.show(...);
```

Hinweis: Der Popufehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus

.setActive() ist Teilmenge von .focus(): nur das aktiv setzen

funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.

Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.

Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert



Die Sprechblase erscheint auch dann, wenn das Control mit `style.visibility='hidden'` belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit `Breite == Höhe == 0` gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit `Breite == Höhe == 0` gerendert werden. Wegen Dimensionierung auf 0 sollte `style.visibility="hidden"` sein. Im Falle eines Containers reicht es, den `style` des Containers zu ändern, da `visibility` normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC



<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)
http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen: • http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp (http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen: • <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp> (<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.



Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen:• 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)



Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87='parent.Y_unload(0);'; X86[0]=new Function(",X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet,
so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr
wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
    unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild

<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12) ">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12) ">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

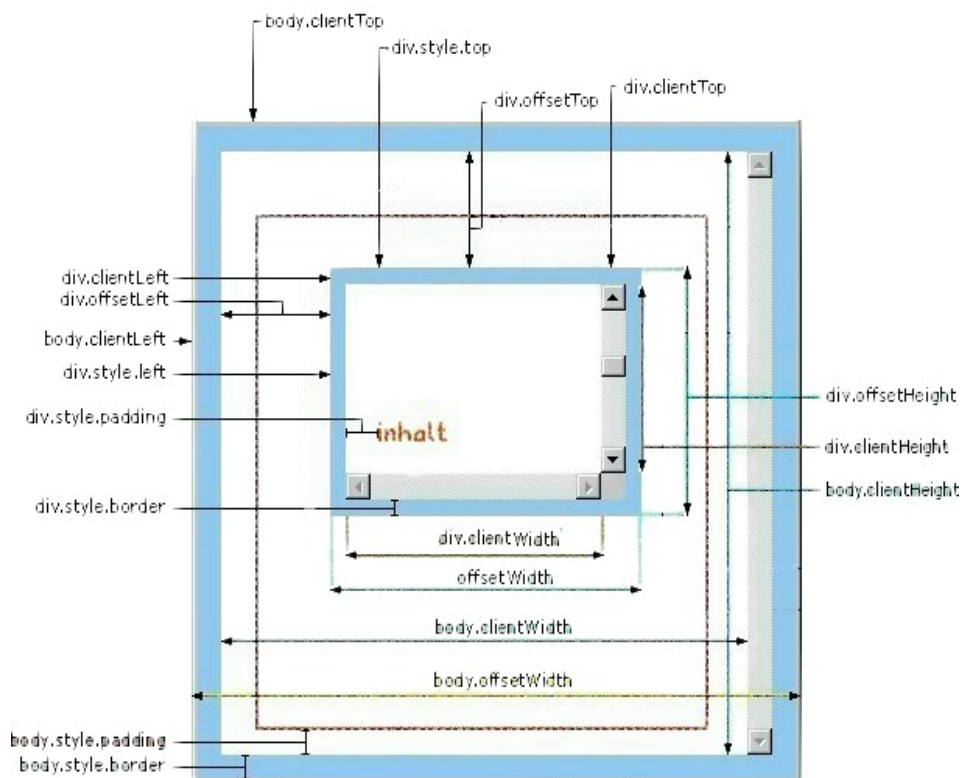
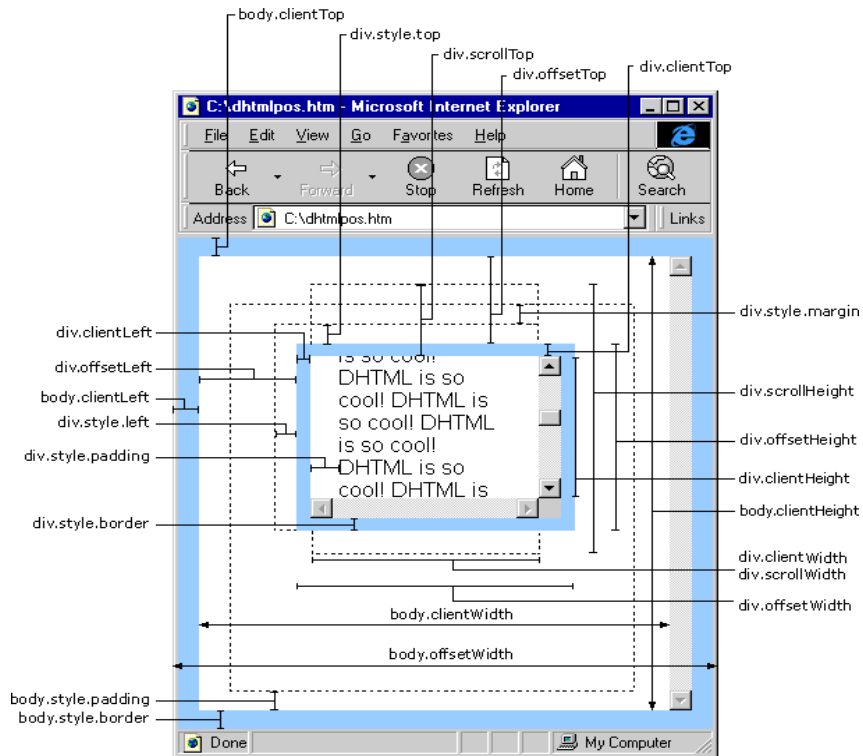
Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,
innerhalb

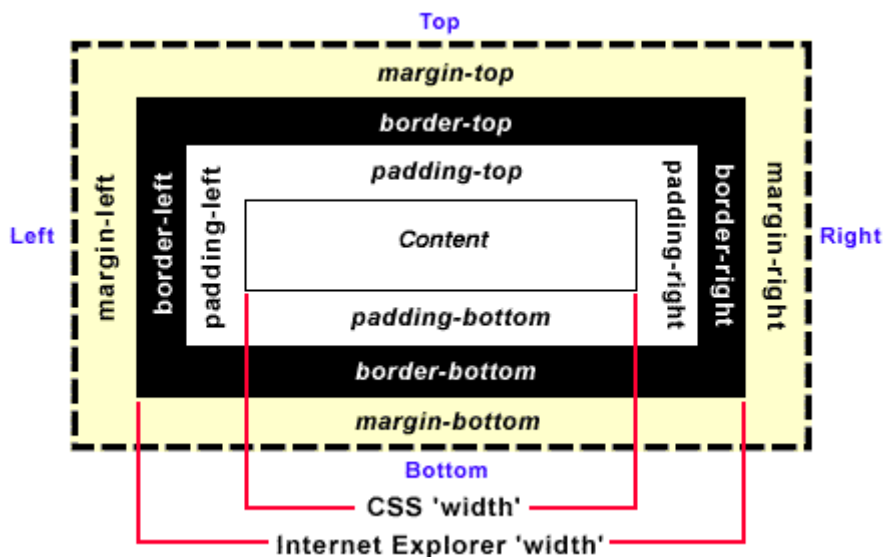
dessen dann die neuen HTML-Elemente erzeugt werden.

Eigenschaften:

Die Eigenschaften des IE als Grafiken:







Hinweis: Falls es sich um das aktuelle Fenster handelt, so kann die Notation `window.eigenschaft` auf `eigenschaft` abgekürzt werden. Innerhalb von Eventhandlern ist immer **window**.eigenschaft zu kodieren, also die volle Referenzierung. Anstelle von `window` kann auch `self` kodiert werden. Theoretisch ist auch `with (window) { }` kodierbar.

<code>.clientInformation</code>	Zeiger auf das Objekt <code>window.clientInformation</code> , welches vom Objekt <code>navigator</code> erbt
<code>.clipboardData</code>	Zeiger auf Objekt <code>clipboardData</code> als Windows-Zwischenablage zu verwenden mit Eventhandlern <code>onbeforecut</code> und <code>onbeforepaste</code> siehe auch <code>event.dataTransfer</code> Objekt
<code>.closed</code>	Zustand auf Geschlossenheit eines Fensters Syntax: <div style="margin-left: 40px;"> <code>[var Wert =] logischer_window_name.closed</code> <div style="display: flex; justify-content: space-between;"> Wert <code>false</code> so ist Fenster offen <code>true</code> so ist Fenster geschlossen </div> </div> <div style="margin-left: 40px;"> <code>logischer_window_name</code> Zeiger laut <code>open()</code> nur lesen </div>
<code>.defaultStatus</code>	Standard-Text der Statuszeile (nicht der aktuelle Text) siehe <code>.status</code> Verwendung in Eventhandler-Funktion: Es muss return true; kodiert werden Syntax: <div style="margin-left: 40px;"> <code>logischer_window_name.defaultStatus = Kette</code> <code>[var Kette =] logischer_window_name.defaultStatus</code> <div style="display: flex; justify-content: space-between;"> Kette String </div> </div> <div style="margin-left: 40px;"> <code>logischer_window_name</code> Zeiger laut <code>open()</code> lesen und schreiben </div>
<code>.dialogArguments</code>	Zeiger auf Objekt <code>window.dialogArguments</code> als Argumente für modale oder nicht modale Dialoge ab IE 5.x wird per <code>showModalDialog()</code> bzw. <code>showModelessDialog()</code> instanziiert
Beispiel:	<pre> <HTML> <HEAD> <SCRIPT> function Anzeige() { // Zeiger auf die Formularelemente holen var FormularElemente = ID_Formular.elements; // Formulardaten in einem privaten Objekt kapseln als Argument für modalen Dialog var Formulardaten = new Object(); Formulardaten.firstName = FormularElemente.ID_Input1.value; </pre>



```

FormularDaten.lastName = FormularElemente.ID_Input2.value;

// Fenster als modalen Dialog anzeigen und dabei Eigenschaft .dialogArguments füllen
//     Mit der Übergabe der Daten erfolgt das Öffnen des neuen Fenster, das
//         sich auf die namensgleichen Eigenschaften von FormularDaten
//         beruft, deren Bezeichner mit in den Argumenten übergeben werden
window.showModalDialog( "test.htm",
                        FormularDaten,
                        "dialogHeight:300px; dialogLeft:200px;"
                      );
}
</SCRIPT>
</HEAD>
<BODY>
  <FORM ID= "ID_Formular">
    Vorname:
    <INPUT TYPE="text" NAME="ID_Input1" VALUE="Vorname">
    <BR>
    Nachname:
    <INPUT TYPE="text" NAME="ID_Input2" VALUE="Nachname">
  </FORM>
  <BR>
  <BUTTON onclick="Anzeige();" >Formulardaten im modalen Dialog anzeigen</BUTTON>
</BODY>
</HTML>

```

test.htm enthält:

```
<HTML>
<HEAD>
<SCRIPT>
    function Anzeigen()
    {
        var DialogArgumente = window.dialogArguments;

        // Es müssen namensidentische Bezeichner der Eigenschaften von
        //     DialogArgumente verwendet werden:
        //         firstName und lastName werden in der
        //             aufrufenden Webseite definiert
        //             und müssen hier ebenfalls verwendet werden
        document.writeln("Vorname = " + DialogArgumente.firstName);
        document.write("Nachname = " + DialogArgumente.lastName);
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeigen();">
</BODY>
</HTML>
```

<code>.dialogHeight</code>	Höhe des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
	ab IE 5.x
	Syntax:
	<code>logischer_window_name..dialogHeight [= Kette]</code> <code>[var Kette =] logischer_window_name..dialogHeight</code>
	Kette
	String als numerisches Literal mit kodierter Einheit <code>"cm", "mm", "in", "pt", "pc"</code> oder <code>"px"</code>
	wenn < 100 Pixel so 100 Pixel automatisch verwendet
	<code>logischer_window_name</code>
	Zeiger laut <code>open()</code>
	lesen und schreiben

Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
```



```
event.srcElement.blur(); // Focus dem QuellElement wegnehmen
window.showModalDialog("test.htm",
    "",
    "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
);
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>
```

<code>.dialogLeft</code>	X-Koordinate der linken oberen Ecke des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
ab IE 5.x	
Syntax:	
	<code>logischer_window_name..dialogLeft [= Kette]</code> <code>[var Kette =] logischer_window_name..dialogLeft</code>
	Kette
	String als numerisches Literal mit kodierter Einheit "cm", "mm", "in", "pt", "pc" oder "px"
	wenn < 100 Pixel so 100 Pixel automatisch verwendet
	logischer_window_name
lesen und schreiben	Zeiger laut open()

Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>
```

<code>.dialogTop</code>	Y-Koordinate der linken oberen Ecke des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
ab IE 5.x	
Syntax:	
	<code>logischer_window_name..dialogTop [= Kette]</code> <code>[var Kette =] logischer_window_name..dialogTop</code>
	Kette
	String als numerisches Literal mit kodierter Einheit "cm", "mm", "in", "pt", "pc" oder "px"
	wenn < 100 Pixel so 100 Pixel automatisch verwendet
	logischer_window_name
lesen und schreiben	Zeiger laut open()



Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>
```

<code>.dialogWidth</code>	Breite des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
	ab IE 5.x
	Syntax:
	<code>logischer_window_name..dialogWidth [= Kette]</code> <code>[var Kette =] logischer_window_name..dialogWidth</code>
	Kette
	String als numerisches Literal mit kodierter Einheit "cm", "mm", "in", "pt", "pc" oder "px"
	wenn < 100 Pixel so 100 Pixel automatisch verwendet
	<code>logischer_window_name</code>
	Zeiger laut <code>open()</code>

lesen und schreiben

Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();           // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>
```

.document	Zeiger auf das Objekt document im Fenster
.event	Zeiger auf das Objekt event im Fenster
.external	Zeiger auf das Objekt window.external im Fenster
.frameElement	Zeiger auf Frame bzw. IFrame, die im Fenster liegen mit diesem Zeiger können im Fenster alle Eigenschaften und Methoden des Frame bzw. IFrame referenziert werden

Beispiel:

<SCRIPT LANGUAGE="JScript">



```

var FrameZeiger = window.frameElement;
FrameZeiger.src = "http://www.test.de ";
</SCRIPT>

```

.frames Zeiger auf die Collection document.frames im HTML-Dokument, das im Fenster angezeigt wird

.history Zeiger auf das Objekt history (Verlauf)

.length Anzahl aller Frames bzw. IFrames im Fenster laut Collection document.frames

Syntax:

```
[ var Wert = ] logischer_window_name.length
```

Wert

Integer, ab 1

logischer_window_name

Zeiger laut open()

nur lesen

.location Zeiger auf das Objekt location

.name physischer Fenstertitel laut open()

entspricht Wert des Attributes TARGET eines Links

Text des Fenstertitels (Text in Titelzeile des Fensters) laut document.title

Syntax:

```
logischer_window_name.name = Kette
```

```
[ var Kette = ] logischer_window_name.name
```

Kette

String

logischer_window_name

Zeiger laut open()

lesen und schreiben

Beispiel:

```

window.name="MyWindow";
window.open("file.htm","Frame1");

```

.navigator Zeiger auf das Objekt navigator

.offscreenBuffering alle Objekt in aktueller Seite im Offscreen zeichnen bevor sie sichtbar werden

Syntax:

```
logischer_window_name.offscreenBuffering [ = Kette ]
```

```
[ var Kette = ] logischer_window_name.offscreenBuffering
```

Kette

String

"auto"

Default, Browser entscheidet

"true"

offscreen buffering ein

"false"

offscreen buffering aus,

also direkt sichtbar auf
Bildschirm zeichnen

logischer_window_name

Zeiger laut open()

lesen und schreiben

.opener Zeiger auf Elternfenster, das **open()** enthält und das Kind-Fenster öffnet, welches in dieser

.opener-Eigenschaft den Zeiger auf das Elternfenster enthält

Bezug im geöffneten Fenster auf Instanzen des Aufrufers ist möglich

Bezug des Aufrufers auf geöffnetes Fenster ist möglich

Hinweis: Bei FRAME und IFRAME anstelle opener den Zeiger parent verwenden

Öffnet ein Frame / IFrame ein Fenster, das damit nicht im Frameset läuft, so ist der

Bezug vom Fenster aus auf das Frameset oder auf einen Frame im Frameset per

window.opener.parent möglich, solange opener existiert.

Syntax:

```
logischer_window_name.opener [ = Zeiger ]
```

```
[ var Zeiger = ] logischer_window_name.opener
```

logischer_window_name

Zeiger laut open()

lesen und schreiben

Beispiel:

```

function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

```



```

    }

    ....

    // Fenster öffnen
    var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
    var FensterDokumentZeiger = FensterZeiger.document;

    var Kette= '<HTML>'
        + '<HEAD></HEAD>'
        + '<BODY >'
        + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
        + '<BR>'
        + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
        + ' onclick="opener.OnClickHandler();"'
        + '>'
        + '</BODY>'
        + '</HTML>';

    FensterDokumentZeiger.open("text/html");
    FensterDokumentZeiger.write(Kette);
    FensterDokumentZeiger.close();

```

.parent Zeiger auf das **in der Fensterhierarchie übergeordnete** Fenster, das aber nicht das .open() zum Kindfenster enthalten muss
 z.B. Zeiger auf das Fenster, das die FRAMESET-Deklaration enthält,
 oder das diesem Fenster übergeordnet ist
 Test auf Existenz von parent per if (parent != self)
 Syntax:
 [var Zeiger =] logischer_window_name.parent

 logischer_window_name Zeiger laut open()

nur lesen

.returnValue Returnwert des Dialog-Fensters, das mit Methoden
 .showModalDialog() bzw. .showModelessDialog()
 erzeugt wurde
 wird gefüllt durch .showModalDialog() bzw. .showModelessDialog()
 Syntax:
 logischer_window_name.returnValue [= Wert]
 [var Wert =] logischer_window_name.returnValue

 Wert Ausdruck etc.

 logischer_window_name Zeiger laut open()

lesen und schreiben

.screen Zeiger auf das Objekt screen

.screenLeft X-Koordinate der linken oberen Fensterecke (ohne Bar, Menü, Scrollbalken, Statuszeile etc)
 bezüglich linke obere Ecke (0,0) vom Bildschirm
 siehe Objekt window
 Syntax:
 [var Wert =] logischer_window_name.screenLeft

 Wert Integer, in Pixels
 0 = linke obere Ecke des Bildschirms

 logischer_window_name Zeiger laut open()

nur lesen

.screenTop Y-Koordinate der linken oberen Fensterecke (ohne Bar, Menü, Scrollbalken, Statuszeile etc)
 bezüglich linke obere Ecke (0,0) vom Bildschirm
 Syntax:
 [var Wert =] logischer_window_name.screenTop

 Wert Integer, in Pixels
 0 = linke obere Ecke des Bildschirms

 logischer_window_name Zeiger laut open()

nur lesen



Beispiel für automatisch wechselnder Text in der Statuszeile:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var statuszeile = new Array();
    statuszeile[0] = "Text0";
    statuszeile[1] = "Text1";
    statuszeile[2] = "Text2";

    var statuszeile_nr = 0;

    function textanzeigen()
    {
        window.status = statuszeile[statuszeile_nr];
        statuszeile_nr ++;
        if (statuszeile_nr >= statuszeile.length) {statuszeile_nr = 0;}
        setTimeout("textanzeigen()", 1000);
    }
// -->
</SCRIPT>
</HEAD>
<BODY onLoad="textanzeigen()">
</BODY>
</HTML>
```

Beispiel für dauerhaftes Scrollen eines Textes in der Statuszeile:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext        = "";
    var zahler            = scrolltext_gesamt.length;

    function scrollen()
    {
        if (zahler == scrolltext_gesamt.length)
        {
            zahler = 0;
            scrolltext = scrolltext_gesamt;
        }
        else
        {
            zahler++; // 1 bis scrolltext_gesamt.length
            // Blank vorsetzen, also Kette um 1 Zeichen verlängern
            scrolltext = " "+scrolltext;

            // rausgerutschtes Zeichen abschneiden
            scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
            // Angaben ab 0
        }

        window.status = scrolltext;

        setTimeout("scrollen()", 100);
    }
// -->
</SCRIPT>
</HEAD>
<BODY onLoad="scrollen()">
</BODY>
</HTML>
```

Beispiel für einen Scrolltext in der Statusleiste, der angehalten wird, wenn Mauszeiger sich über einen Linkt bewegt:

```
<HTML>
```



```

<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext = "";
    var zahler = scrolltext_gesamt.length;
    var id;

    function scrollen()
    {
        if (zahler >= scrolltext_gesamt.length)
        {
            zahler = 0;
            scrolltext = scrolltext_gesamt;
        }
        else
        {
            zahler++; // 1 bis scrolltext_gesamt.length
            // Blank vorsetzen, also Kette um 1 Zeichen verlängern
            scrolltext = " "+scrolltext;

            // rausgerutschtes Zeichen abschneiden
            scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
            // Angaben ab 0
        }

        window.status = scrolltext;
        id=setTimeout("scrollen()", 100);
    }

    function scrollen_stoppen()
    {clearTimeout(id);}
// -->
</SCRIPT>
</HEAD>

<BODY onLoad="scrollen()">
    Bewegen Sie den Mausfeil &uuml;ber diesen
    <A      HREF="http://www.test.de"
           onMouseOver="scrollen_stoppen()"
           onMouseOut="scrollen()"
    >
    Link
    </A>
</BODY>
</HTML>

```

Beispiel für Anzeige eines Textes zu einem Hyperlink (HREF) für eine feste Zeitspanne in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var id;
    var anzeige_aktiv = false;
    var anzeige_zeit = 3000;      // 3000 Millisekunden = 3 Sekunden

    function anzeige_starten(href_text)
    {
        if(anzeige_aktiv)
        {clearTimeout(id);}

        window.status = href_text;

        id = setTimeout("anzeige_stoppen()", anzeige_zeit);

        anzeige_aktiv = true;

        return true;      // Wichtig !!!
    }

```



```

function anzeige_stoppen ()
{
    anzeige_aktiv = false;

    window.status = "";

}

//-->
</SCRIPT>
</HEAD>
<BODY>
    <A HREF= ... onMouseOver="javascript:return anzeige_starten('Hinweistext...')">
    ...
    </A>
</BODY>
</HTML>

```

Beispiel zur Anzeige eines Formularfeld-Hinweises in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    function hinweis_anzeigen(hinweis_text)
    { window.status = hinweis_text; }

    function hinweis_loeschen()
    { window.status = ""; }

// -->
</SCRIPT>
</HEAD>
<BODY>
    <FORM>
        <INPUT TYPE=TEXT
            onFocus="hinweis_anzeigen('Hinweis');"
            onBlur="hinweis_loeschen();"
        >
    </FORM>
</BODY>
</HTML>

```

Beispiel für blinkende Anzeige mit Zeitspanne von Text in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    var zeitspanne = 6000;        // 6 Sekunden blinken lassen, danach
                                //      Statuszeile löschen
    var anzeigezeit = 500;        // 0,5 Sekunden warten nach Setzen
                                //      bzw. Löschen der Statuszeile
    var id_blinken = null;        // muss auf null initialisiert sein !!

    function blinken_timer_loeschen
    {
        if (id_blinken != null)
        {
            clearTimeout(id_blinken); // blinken stoppen falls aktiv
            id_blinken = null;
        }
    }

    function blinken_stop()
    {
        blinken_timer_loeschen;
        window.status="";
    }

    function blinken_start(status_text)
    {
        blinken_timer_loeschen;
        blinken(true, status_text); // Blinken anstossen für

```



```

                                // paralleles Arbeiten per Timer
        setTimeout("blinken_stop()",zeitspanne); // warten und danach blinken stoppen
    }

    function blinken(ein_aus, text)
    {
        if (ein_aus)
        {
            window.status = text;
            ein_aus=false; // im nächsten Aufruf die Statuszeile löschen
        }
        else
        (
            window.status = "";
            ein_aus=true; // im nächsten Aufruf die Stautuszeile setzen
        )

        id_blinken = setTimeout("blinken(" + ein_aus + ")", anzeigezeit)
                                // nächsten Aufruf nach Ablauf der anzeigezeit starten
    }

//-->
</SCRIPT>
</HEAD>
<BODY ... onLoad="blinken_start('Ich blinke !')">
</BODY>
</HTML>

```

.top Zeiger auf das oberste Fenster in der Fenster-Hierarchie
 siehe Objekt window
 Syntax:
 [var Zeiger =] logischer_window_name.top
 logischer_window_name Zeiger laut open()
 nur lesen

Methoden

Hinweis: Falls es sich um das aktuelle Fenster handelt, so kann die Notation window.methode() auf methode() abgekürzt werden.
 Innerhalb von Eventhandlern ist immer **window.methode()** zu kodieren, also die volle Referenzierung.
 Anstelle von window kann auch self kodiert werden.
 Theoretisch ist auch with (window) { } kodierbar.

.alert() Dialogbox erzeugen
 1. Anzeige von:
 Ausrufungszeichen-Symbol
 variablen String
 OK-Button und
 warten auf Drücken des OK-Button
 Syntax:
 logischer_window_name alert([Kette])
 Kette String oder Stringausdruck
 logischer_window_name Zeiger laut open()

liefert nichts

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
 NICHT verkettet sondern in **Zufallsfolge**, es sei denn
 die Handler prüfen ihre Aufruffolge (muss programmiert werden)
 Vor dem Neuelegen immer den Standard-Eventhandler retten und diesen
 nach .detachEvent() wieder einbinden (siehe Beispiel).
 Syntax:
 [var Wert =] logischer_window_name.attachEvent(Kette, Zeiger)
 Kette String
 Eventbezeichner
 Zeiger auf Eventhandler
 Wert true Einschalten erfolgreich
 false Einschalten nicht möglich gewesen



Beispiel 1: logischer_window_name Zeiger laut open()

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor      = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
        window.onmouseover = Rette_Standard_Eventhandler_OnMouseOver;

        detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
        window.onmouseout = Rette_Standard_Eventhandler_OnMouseOut;
    }

    var Rette_Standard_Eventhandler_OnMouseOver = window.onmouseover;
    attachEvent ('onmouseover', CursorNeu);

    var Rette_Standard_Eventhandler_OnMouseOut = window.onmouseout;
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

Beispiel 2:

```
function ResizeHandler()           // Homepage neu laden
{window.history.go(0);}

// Standardhandler retten
var RetteHandler_Resize = window.onresize;

// und neu belegen
window.onresize = ResizeHandler;    // Homepage neu laden
                                   // ohne () kodieren, damit nicht sofort ausgeführt wird

// und Start des Abfangens vom resize-Event
window.attachEvent('onresize', ResizeHandler);

.....

// irgendwann abschalten der Eventüberwachung
window.detachEvent('onresize', ResizeHandler);

// und Standardhandler wieder einbinden
window.onresize = RetteHandler_Resize;
```

.blur() Element den Focus wegnehmen und Event onblur auslösen
 Der Focus wird **nicht automatisch** auf irgend ein anderes Element gesetzt !
 vor IE 5.0 TABINDEX-Attribut muss kodiert sein
 ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
 Syntax: logischer_window_name.blur()

logischer_window_name Zeiger laut open()
 liefert nichts



<code>.clearInterval()</code>	stoppt einen Timer, der mit <code>.setInterval()</code> gestartet wurde	
	Syntax:	
	<code>logischer_window_name</code>	<code>.clearInterval(timer_id)</code>
	<code>timer_id</code>	Integer laut Rückgabewert von <code>.setInterval()</code>
	<code>logischer_window_name</code>	Zeiger laut <code>open()</code>
	liefert nichts	

Beispiel 1:

```
var timerID = null;

function timer()
{
    // tue was
}

function starttimer()
{
    if (timerID == null)
        {timerID = setInterval("timer()", 1000);}
}

function stoptimer()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
    }
}
```

Beispiel 2 für Sekundenbalken:

```
<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekudentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;
        }
    }
</SCRIPT>
</HTML>
```



```

        // Stop-Button aktivierbar machen
        ID_Button2.disabled = false;

        // Uhr neu starten
        timerID = setInterval("Uhr()", 1000);
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

    function ZurueckSetzen()
    { Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

.clearTimeout() löscht ein Timeout, das mit **.setTimeout()** gesetzt wurde
 Syntax:

```

        logischer_window_name .clearTimeout(timeout_id)

        timeout_id                Integer
                                   laut Rückgabewert von .setTimeout()

        logischer_window_name      Zeiger laut open()

    liefert nichts

```

.close() Fenster schliessen, das offen ist
 Fenster muss nicht explizit mit Methode **.open()** Methode erzeugt worden sein
 Schliessen des letzten Browserfensters erzeugt immer Dialog-Box-Abfrage
 Fenster des Dokumentes schliessen: **document.close()**
 innerhalb von Eventhandler: Immer **window.close()** oder **logischer_window_name.close()** oder **self.close()**
 kodieren
 Syntax:

```

        logischer_window_name.close()

```



logischer_window_name Zeiger laut open()
 liefert nichts

Beispiel:

```
<BODY onclick="window.close();">
  Klick zum Schliessen des Fensters
</BODY>
```

Beispiel für Fenster schliesst sich selbst nach Wartezeit:

```
</HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
  function fenster_offnen_und_schliessen()
  {

      var fenster;
      fenster=window.open("", "Fenster", "width=180,height=100");
      fenster.document.write("<H1>Ich schlieÙe mich nach 4 Sekunden</H1>");
      fenster.setTimeout('window.close()',4000);

  }

  //-->
</SCRIPT>
</HEAD>
<BODY onload="fenster_offnen_und_schliessen()">
</BODY>
</HTML>
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ ' onclick="opener.OnClickHandler();"'
+ '>'
+ '</BODY>'
+ '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.confirm()

Dialogbox erzeugen mit

1. Anzeige Fragezeichen, String, OK/Abbrechen-Button
2. warten auf Drücken eines der beiden Button

Syntax:

```
[ var Wert = ] logischer_window_name.confirm([Kette])
```

Kette String oder Stringausdruck

Wert true für OK
 false für Abbrechen

logischer_window_name Zeiger laut open()

.createPopup()

Popupfenster ohne irgendwelche Fensterelemente öffnen z.B. für Anzeige von Tooltips
 siehe Objekt window.popup



Fenster wird mit der Anzeige per Methode `.show()` zum aktuellen Fenster erst geschlossen, wenn **anderes Fenster** aktuell wird
z.B. durch Klicken außerhalb des Popupfensters

ab IE 5.5

Syntax:

```
[ var Zeiger = ] logischer_window_name.createPopup()
```

Zeiger

Referenz auf das Popupfenster (Zeiger entspricht ID),
wird für die Verwendung der Eigenschaften und Methoden benutzt per
Zeiger.eigenschaft
Zeiger.methode()

logischer_window_name

Zeiger laut open()

Beispiel

```
var PopUpID=window.createPopup();
PopUpID.document.body.style.backgroundColor="green";
PopUpID.document.body.innerHTML="TooltipText";
PopUpID.show(100,100,180,25,document.body);
```

`.detachEvent()`

Abschalten des Registrieren eines Events durch Eventhandler, wobei Registrierung mit Methode `.attachEvent()` aktiviert wurde

Achtung: Vor dem Neuhelegen des Standard-Eventhandler diesen vor `.attachEvent()` retten und den Standard-Eventhandler **nach** `.detachEvent()` wieder einbinden (siehe Beispiel).

Syntax:

```
logischer_window_name.detachEvent(Kette, Zeiger)
```

Kette

String mit Eventbezeichner

Zeiger

Zeiger auf **denselben** Eventhandler laut `.attachEvent()`

logischer_window_name

Zeiger laut open()

liefert nichts

Beispiel 1:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>
```

```
<SCRIPT LANGUAGE="JScript">
```

```
function CursorNeu()
```

```
{
```

```
    if (event.srcElement == element)
```

```
    {
```

```
        normalColor = style.color;
```

```
        runtimeStyle.color = "red";
```

```
        runtimeStyle.cursor = "hand";
```

```
    }
```

```
}
```

```
function CursorNormal()
```

```
{
```

```
    if (event.srcElement == element)
```

```
    {
```

```
        runtimeStyle.color = normalColor;
```

```
        runtimeStyle.cursor = "";
```

```
    }
```

```
}
```

```
function EventEntfernen()
```

```
{
```

```
    detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
```

```
    window.onmouseover = Rette_Standard_Eventhandler_OnMouseOver;
```

```
    detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
```

```
    window.onmouseout = Rette_Standard_Eventhandler_OnMouseOut;
```

```
}
```

```
var Rette_Standard_Eventhandler_OnMouseOver = window.onmouseover;
```

```
attachEvent ('onmouseover', CursorNeu);
```

```
var Rette_Standard_Eventhandler_OnMouseOut = window.onmouseout;
```

```
attachEvent ('onmouseout', CursorNormal);
```



</SCRIPT>

Beispiel 2:

```

function ResizeHandler()           // Homepage neu laden
{ window.history.go(0); }

// Standardhandler retten
var RetteHandler_Resize = window.onresize;

// und neu belegen
window.onresize = ResizeHandler;   // Homepage neu laden
                                   // ohne () kodieren, damit nicht sofort ausgeführt wird

// und Start des Abfangens vom resize-Event
window.attachEvent('onresize', ResizeHandler);

.....

// irgendwann abschalten der Eventüberwachung
window.detachEvent('onresize', ResizeHandler);

// und Standardhandler wieder einbinden
window.onresize = RetteHandler_Resize;

```

.execScript()

Pendant zur Methode .eval()
 Script wird sofort ausgeführt
 Script in diversen Sprachen möglich
 alle vorhandenen-globalen Variablen ansprechbar
 Syntax:

```
[ var Zeiger = ] logischer_window_name.execScript(Kette1 [, Kette2])
```

Kette1	String mit Scriptcode
Kette2	String mit Scriptsprache Standard ist "JScript"
Zeiger	immer null (nicht numerisch Null !!)
logischer_window_name	Zeiger laut open()

Beispiel.: window.execScript('alert("Hallo");', 'JScript')

.focus()

Focus setzen und Event onfocus auslösen
 nur nach dem kompletten Laden des Dokumentes
 vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
 Syntax:

```
logischer_window_name.focus()
```

logischer_window_name	Zeiger laut open()
liefert nichts	

.moveBy()

linke obere Fenster-Ecke um Pixeldifferenz auf dem Bildschirm verschieben
 Fenster ganz aus dem BS verschieben ist möglich
 Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Bildschirms
 nicht anwendbar bei Dialog-Fenster: Dafür dialogHeight, dialogWidth, dialogTop und dialogLeft

verwenden.

Syntax:

```
logischer_window_name.moveBy(x, y)
```

x	X-Koordinatendifferenz Integer wenn positiv so Verschiebung nach rechts wenn negativ so Verschiebung nach links
y	Y-Koordinatendifferenz Integer wenn positiv so Verschiebung nach unten wenn negativ so Verschiebung nach oben

logischer_window_name	Zeiger laut open()
liefert nichts	



Beispiel für Fenster des Browser rausschieben und danach neu anzeigen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
var BrowserFenster_AktuelleBreite=2000; // sollte größer als max. übliche Auflösung sein
var BrowserFenster_AktuelleHoehe=2000; // sollte größer als max. übliche Auflösung sein

function moveWin()
{
    for (var i = 1; i < BrowserFenster_AktuelleBreite; i++)
    {window.moveBy(1, 1);}

    window.moveBy((-1)* BrowserFenster_AktuelleBreite,(-1) * BrowserFenster_AktuelleHoehe);
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="moveWin();">
</BODY>
</HTML>
```

`.moveTo()` linke obere Fenster-Ecke laut Pixel-Position auf dem Bildschirm positionieren
Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Bildschirms
nicht anwendbar bei Dialog-Fenster: Dafür `dialogHeight`, `dialogWidth`, `dialogTop` und `dialogLeft`
verwenden.

Syntax:

```
logischer_window_name.moveTo(x, y)
```

x

X-Koordinate
Integer, >=0

y

Y-Koordinate
Integer, >= 0

logischer_window_name

Zeiger laut `open()`

liefert nichts

`.navigate()` lädt neues HTML-Dokument laut Url in das Fenster
entspricht `location.href = "...."`

Syntax:

```
logischer_window_name.navigate(Kette)
```

Kette

String mit Url

logischer_window_name

Zeiger laut `open()`

liefert nichts

Beispiele: `navigate("index.html");`
`navigate ("file:///c:/index.html");`

`.open()` neues Fenster als unterstes der aktuellen Fensterhierarchie erzeugen
und dann oberstes sichtbare Fenster öffnen (anzeigen, rendern und als aktuell setzen)
nicht möglich bei Dialog-Fenster oder Popup-Fenster
Hinweis: Zeiger des Fensters, das `.open()` ausführt, liegt in der Eigenschaft `.opener`
des neu erzeugten Fensters

Syntax:

```
[ var logischer_window_name_neues_fenster = ] logischer_window_name.open(
    [URL]
    [, physischer_fenster_name]
    [, Features]
    [, Replace]
)
```

logischer_window_name

Zeiger laut `open()`
muss Referenz per **window** oder **self**
sein, wenn es sich um das erste
Fenster seit dem Laden des
HTML-Dokumentes handelt
innerhalb von Eventhandler muss immer
volle Referenzierung kodiert



werden z.B.

window.open()
anstelle von .open()

logischer_window_name_neues_fenster Zeiger auf das neue Fenster

URL String mit Url des Dokumentes, das nach dem Öffnen in das
Fenster geladen wird
Standard ist about:blank
kann Leerkette sein, also kein Dokument laden
Daten per '?' + escape() **nicht** übergebbar

physischer_fenster_name dient zur Referenzierung
von einem HTML-Elementen mit dem
Attribut TARGET (Wert des Attributes)
auf das Fenster dienen
z.B. im Link

null null-Zeiger, also keine
Referenzierung möglich

String Titel des Fensters
oder vordefinierter Namen
Standard ist "_blank"
also ohne Name

vordefinierte Namen:

"_blank"	Standard : ohne Name
"_media"	Dokument laut Url wird in den HTML-Content-Bereich der Media Bar geladen
"_parent"	ab IE 6.x Dokument laut Url wird in den Elternframe geladen wenn kein Frame so wirkunglos
"_search"	Dokument laut Url wird in das Browser-Search- Fenster geladen
"_self"	ab IE 5.x Dokument laut Url wird in das neue Fenster geladen
"_top"	Dokument laut Url wird in das oberste Fenster der Fensterhierarchie geladen

Features String als Parameterliste mit Kommatrennung
(Liste der Fensterkomponenten)
gesamte Liste in " " bzw. ' ' setzen
z.B. "fullscreen=yes, toolbar=yes"
gesamte Liste ist 1 Zeichenkette,
die in 1 Quelltextzeile passen muss, oder in Teilketten zerlegt
mit dem + Operator zusammengesetzt wird
Blanks in Liste **nicht** zulässig

Listenelement: option=wert
wenn mindestens 1 Element kodiert, so alle
anderen nicht kodierten Elemente
automatisch deaktiviert, also immer alle
gewünschten Optionen kodieren !!!
Standardwert eines Merkmals, wenn
Liste kodiert wurde: no oder 0
Liste nicht kodiert wurde: yes oder 1
keine Standardwerte vorhanden für Pixelangaben
da diese vom Browser automatisch
belegbar sind

channelmode = { yes | no | 1 | 0 }



default ist no bzw. 0
yes oder 1 für Channelleiste anzeigen
(Fenster im Channel-Mode anzeigen)

directories = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes oder 1 Directory Buttons anzeigen

fullscreen = { yes | no | 1 | 0 }
default ist no bzw. 0
yes bzw. 1 Vollbild anzeigen also ohne
Leisten etc., wobei damit fast alle
Steuerungselemente unsichtbar werden
Hinweis: ALT+F4 schliesst das Fenster

height = Pixelwert, Fensterhöhe
mindestens 100 (wenn < 100, so auf
100 automatisch gesetzt)

left = X-Koordinate in Pixels bezüglich linker
oberer Screen-Ecke (0,0)

location = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw 1 für URL-Eingabezeile anzeigen
(Adresszeile anzeigen)

menubar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Menübar anzeigen
(Leiste der Pull-Down-Menüs anzeigen)

resizable = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Größenveränderung des
Fensters erlaubt per Mausziehen

scrollbars = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Scrollbalken anzeigen
sobald Fensterinhalt größer als
Anzeigebereich des Fensters

status = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Statuszeile anzeigen

titlebar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Titelzeile anzeigen
Titel werden immer angezeigt bei Dialog-Box

toolbar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Toolbar (Navigationsleiste)
anzeigen (Button Zurück etc.)

top = Y-Koordinate in Pixels bezüglich linker oberer
Screen-Ecke (0,0)

width = Pixelwert, Fensterbreite,
mindestens 100 (wenn < 100, so auf
100 automatisch gesetzt)

Replace true, so History-Eintrag des Dokumentes, das das neue
Fenster erzeugt, ersetzen durch den Eintrag
des neuen Fensters,
also Zurück zum Dokument, dass das Fenster öffnet,
nicht mehr möglich (**Achtung: Diese Einstellung
ist absolut User-unfreundlich, da der User**



nicht das Button "Zurück" verwenden kann, sondern erst in der Verlauf -Leiste suchen muss, und somit den Eindruck bekommt, als würde man ihn an die neue Webseite zwanghaft binden wollen ! Das gilt vorallem dann, wenn der User in den Browser-Einstellungen das Öffnen im selben Fenster gewählt hat.)

false, so neuen History-Eintrag zum neuen Fenster erzeugen,
also zurück zum alten Fenster möglich

Beispiel:

```
window.open("Sample.htm",null, "height=200,width=400,status=yes,toolbar=no,menubar=no,location=no");
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;
```

```
var Kette= '<HTML>'
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ 'onclick="opener.OnClickHandler();"
+ '>'
+ '</BODY>'
+ '</HTML>';
```

```
FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.print() ruft Dialog-Box-Druckfenster auf zum Druck des Dokumentes im aktuellen Fenster
entspricht Druckbutton oder Datei-Menü-Drucken
löst folgende Ereignisse aus: onbeforeprint und onafterprint
Syntax:

```
logischer_window_name.print()
```

logischer_window_name Zeiger laut open()

liefert nichts

Beispiel für Verwendung der Ereignisse per Handler

```
onbeforeprint Handler blendet Teile der Seite ein/aus, die gedruckt werden sollen
onafterprint Handler hebt Veränderungen von onbeforeprint wieder auf
```

Beispiel:

```
<INPUT TYPE="button" VALUE="Drucken"onclick="javascript:self.print()">
```

.prompt()

Eingabefenster erzeugen:

1. Meldungstext anzeigen,
Eingabezeile vorbelegen und anzeigen,
OK- bzw. CANCEL-Button anzeigen
auf User.Eingabe in die Zeile und anschliessendem Druck auf OK warten
- 2.

Syntax:

```
[ var Wert = ] logischer_window_name ([Kette1 [,Kette2])]
```

Kette1

String

freier Text der Meldung

Kette2

String,

frei Text der Vorbelegung der Eingabezeile
Standard ist "undefined"



	Wert	String oder Integer Ergebnis der User-Eingabe
	logischer_window_name	Zeiger laut open()
.resizeBy()	Fenstergröße um Pixeldifferenz verändern Fenstergröße auf weniger als 100 x 100 Pixel ist nicht möglich nicht bei Dialog-Fenster: Dafür dialogHeight, dialogWidth, dialogTop, dialogLeft benutzen funktioniert nur, wenn open-Merkmal resizable=yes kodiert wurde Syntax: logischer_window_name.resizeBy(x,y)	
	x	horizontale Spanne in Pixel, Integer wenn positiv so Veränderung nach rechts wenn negativ so Veränderung nach links
	y	vertikale Spanne in Pixel Integer wenn positiv so Veränderung nach unten wenn negativ so Veränderung nach oben
	logischer_window_name	Zeiger laut open()
	liefert nichts	
.resizeTo()	Fenstergröße neu dimensionieren in Pixel Fenstergröße auf weniger als 100 x 100 Pixel ist nicht möglich nicht bei Dialog-Fenster: Dafür dialogHeight, dialogWidth, dialogTop, dialogLeft benutzen funktioniert nur, wenn open-Merkmal resizable=yes kodiert wurde Syntax: logischer_window_name.resizeTo(x, y)	
	x	Breite in Pixel, Integer >=100
	y	Höhe in Pixel, Integer >=100
	logischer_window_name	Zeiger laut open()
	liefert nichts	

Beispiel für Fensterauflösung ändern:

```
javascript:window.resizeTo(640,480); javascript:window.resizeTo(800,600); javascript:window.resizeTo(1024,768)
```

Beispiel für sich aufblasendes Fenster von 100x100 bis auf 640x480

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    var start_hoehe=100;
    var start_breite=100;
    var max_hoehe=480;
    var max_breite=640;
    var aktuelle_hoehe=start_hoehe;
    var aktuelle_breite=start_breite;
    var y=5;
    var TimerID=null;
    var fenster;

    function start()
    {
        fenster=window.open("", "", "scrollbars");

        if ( document.layers || document.all)
        {
            fenster.resizeTo(start_breite,start_hoehe);
            fenster.moveTo(0,0);
            blasen();
        }
        else
        {alert("Weder Netscape noch Microsoft erkannt !")};
    }
-->
</HTML>
```



```

function blasen()
{
    if (aktuelle_hoehe>=max_hoehe)
    {x=0;}

    fenster.resizeBy(5,y);
    aktuelle_hoehe+=5;
    aktuelle_breite+=5;

    if (aktuelle_breite>=max_breite)
    {
        alert("Maximal aufgeblasen !");
        fenster.close();
        x=5;
        TimerID=null;
    }
    else
    {TimerID=setTimeout("blasen()",50);}
}
//-->
</SCRIPT>
</HEAD>
<BODY>
<A    HREF="javascript:start()"
      onMouseOver="javascript:window.status='Oeffne Fenster';return true;" // Text nach Statuszeile
      onMouseout="javascript:window.status=";" // Statuszeile löschen
>Oeffne NEUES Fenster mit 100x100 Pixel und blase es auf 640x480 Pixel !
</A>
</BODY>
</HTML>

```

.scroll() deprecated und zu ersetzen durch **.scrollBy()** bzw. **.scrollTo()**
linke obere Ecke des Dokumentes im Fenster verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters
Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrollleisten etc.
sinnvoll, wenn automatische Anzeige von Scrollleisten verhindert wurde
Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters
Syntax:

logischer_window_name.scroll(x, y)	
x	Pixel-Abstand vom linken Fensterrand Integer, > 0
y	Pixel-Abstand vom oberen Fensterrand Integer, > 0
logischer_window_name	Zeiger laut open()

liefert nichts

.scrollBy() ersetzt **.scroll()**
linke obere Ecke des Dokumentes im Fenster verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters
Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrollleisten etc.
sinnvoll, wenn automatische Anzeige von Scrollleisten verhindert wurde
Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters
Syntax:

logischer_window_name.scrollBy(x, y)	
x	Pixel-Abstand vom linken Fensterrand Integer wenn positiv, so Verschiebung nach rechts wenn negativ, so Verschiebung nach links
y	Pixel-Abstand vom oberen Fensterrand Integer wenn positiv, so Verschiebung nach unten wenn negativ, so Verschiebung nach oben



	logischer_window_name	Zeiger laut open()
	liefert nichts	
.scrollTo()	ersetzt .scroll() linke obere Ecke des Dokumentes im Fenster auf Pixelposition bezüglich linker oberer Ecke des Anzeigebereiches des Fensters setzen Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrollleisten etc. sinnvoll, wenn automatische Anzeige von Scrollleisten verhindert wurde Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters Syntax:	
	logischer_window_name.scrollTo(x, y)	
	x	X-Koordinate Integer, >= 0
	y	Y-Koordinate Integer, >=0
	logischer_window_name	Zeiger laut open()
	liefert nichts	
.setActive()	Fenster als aktiv setzen (also auch für die Eventdurchreichung aktivieren) aber ohne es zu fokussieren und ohne es scrollbar zu machen Syntax:	
	logischer_window_name.setActive()	
	logischer_window_name	Zeiger laut open()
	liefert nichts	
Beispiel	<pre> <HTML> <HEAD> <SCRIPT> var ID_Fenster; function FensterErzeugen() { ID_Fenster = window.open("/test /test.htm", "ID_Fenster", "top=10px,left=480px,height=375px,width=200px,resizable=1"); this.focus(); } function ButtonAktivieren() {window.parent.ID_Fenster.ID_Button.setActive();} </SCRIPT> </HEAD> <BODY onload="FensterErzeugen();"> <BUTTON ID="ID_Button" onclick="ButtonAktivieren();"> Button aktivieren </BUTTON> </BODY> </HTML> </pre>	
.setInterval()	endlos-periodischer Aufruf eines Codes mit jeweiligem vorherigen Warten in Millisekunden (Timer) (getimte Rekursion) Der mit setInterval() aufgerufene Code wird zyklisch aktiviert, wobei nach dem ERSTEN Aufruf von setInterval() mit der Folgeanweisung hinter .setInterval() sofort weitergemacht wird, während die Rekursion parallel läuft. Damit gilt: setInterval() kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da sonst diese Anweisungen während der parallelen rekursiven Ereignisüberwachung bereits abgearbeitet werden. Syntax:	
	[var TimerID =] logischer_window_name.setInterval(Code, Wartezeit [, Sprache])	
	Code	auszuführender Code bzw. Zeiger auf Codebaustein vor IE 5.x: String aber kein Zeiger ab IE 5.x: String oder Zeiger Zeiger empfehlenswert, wenn Code in externer Datei



liegt
String empfehlenswert, wenn Code im aktuellen
Dokument liegt
Übergabe von Argumenten möglich

Wartezeit

Wartezeit nach deren Ablauf wird Code aktiviert
wird

Integer
in Millisekunden
(1000 Millisekunden sind 1 Sekunde)

Sprache

Sprache des Codes
(analog zum LANGUAGE-Attribut)

String
z.B. "JScript", "VBScript", "JavaScript"

TimerID

ID für den Stop der periodischen Aufruffolge mit
der Methode .clearInterval()

Integer

logischer_window_name

Zeiger laut open()

Beispiel 1

String

```
window.setInterval("EineFunktion()", 5000);
```

Zeiger

```
window.setInterval(EineFunktion, 5000); // ohne () kodieren
```

Beispiel 2

```
function callback1(){alert("callback1");}
```

```
function callback2(){alert("callback2");}
```

```
function chooseCallback(Nummer)
```

```
{
```

```
    switch (Nummer)
```

```
    {
```

```
        case 0: return callback1;
```

```
        case 1: return callback2;
```

```
        default: return "";
```

```
    }
```

```
}
```

```
var Nummer = 1;
```

```
window.setInterval(chooseCallback(Nummer), 5000);
```

Beispiel 3

```
var SekundenZahler=0;
```

```
var timer_id=null;
```

```
function ZyklischeAktion()
```

```
{
```

```
    SekundenZahler++;
```

```
    window.status= SekundenZahler + " Sekunden ";
```

```
    if ( ( SekundenZahler >= 60 )
```

```
        && ( timer_id != null)
```

```
    )
```

```
    {window.clearInterval(timer_id);}
```

```
}
```

```
timer_id=window.setInterval(ZyklischeAktion,1000);
```

Beispiel 4

```
var SekundenZahler=0;
```

```
var timer_id=window.setInterval("window.status= SekundenZahler++",1000);
```

Beispiel 5:

```
var timerID = null;
```

```
function timer()
```

```
{
```

```
    // tue was
```



```

    }

    function starttimer()
    {
        if (timerID == null)
        {timerID = setInterval("timer()", 1000);}
    }

    function stoptimer()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
        }
    }

```

Beispiel 6 für Sekundenbalken:

```

<HTML>
<HEAD>
<STYLE>
    BUTTON { font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekudentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

```



```

    }
}

function ZurueckSetzen()
{
    Zahler = 0;
}
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

`.setTimeout()` einmaliger und somit **nicht periodischer** Aufruf eines Codes mit genau einem vorherigen Warten in Millisekunden (Timer)

Der mit `setTimeout()` aufgerufene Code wird **nicht zyklisch** aktiviert, wobei nach dem Aufruf von `setTimeout()` mit der Folgeanweisung hinter `setTimeout()` sofort weitergemacht wird. Damit gilt: `setTimeout()` kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da diese Anweisungen bereits während der Ereignisüberwachung abgearbeitet werden.

Syntax:

```
[ var TimerID = ] logischer_window_name.setTimeout(Code, Wartezeit [, Sprache])
```

Code	auszuführender Code bzw. Zeiger auf Codebaustein vor IE 5.x: String aber kein Zeiger ab IE 5.x: String oder Zeiger Zeiger empfehlenswert, wenn Code in externer Datei liegt String empfehlenswert, wenn Code im aktuellen Dokument liegt Übergabe von Argumenten möglich
Wartezeit	Wartezeit nach deren Ablauf wird Code aktiviert wird Integer in Millisekunden (1000 Millisekunden sind 1 Sekunde)
Sprache	Sprache des Codes (analog zum LANGUAGE-Attribut) String z.B. "JScript", "VBScript", "JavaScript"
TimerID	ID für den Stopp der periodischen Aufruffolge mit der Methode <code>.clearTimeout()</code> Integer wird nur benötigt, wenn der Code auf eine rekursive Funktion weist, wobei der Zyklusabbruch



durch `clearTimeout()` erfolgen muss
Hinweis: Ist keine Rekursion nötig, soll aber
zyklisch aufgerufen werden, dann
`.setInterval()` verwenden

```

logischer_window_name    Zeiger laut open()
Beispiel 1
    window.setTimeout("alert('Hallo')", 1000);

```

```
Beispiel 2
var Text = "Hallo ";
window.setTimeout(    "alert("
                        + Text
                        + ")",
                        1000
                    );
```

```
Beispiel 3
<SCRIPT>
    function Start(ZeigerAufObjekt)
    { window.setTimeout("Kode(" + ZeigerAufObjekt.id + ")", 3000);}

    function Kode(ID)
    {
        var Zeiger = eval(ID);
        Zeiger.style.display="none";
    }
</SCRIPT>
<INPUT TYPE=button VALUE="Unsichtbar in 3 Sekunden"
      ID="ID_Button" onclick=" Start(this)"
```

```
Beispiel 4
var SekundenZahler=0;
var timeout_id=null;

function ZyklischeAktion()
{
    SekundenZahler++;
    window.status= SekundenZahler + " Sekunden ";

    if (      ( SekundenZahler >= 60 )
        && ( timeout_id != null )
        )
        {window.clearTimeout(timeout_id);}
}

timeout_id=window.setTimeout(ZyklischeAktion,1000);
```

```
var timeout_id=window.setTimeout("window.status= 'Es ist 1 Sekunde vergangen !'",1000);
```

<code>.showHelp()</code>	Helpdatei anzeigen (*.chm und *.htm) Syntax: <div style="text-align: center;"> <code>logischer_window_name.showHelp(URL [, ContextID])</code> </div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;"> URL ContextID </div> <div style="text-align: center;"> String, URL der Help-Datei String oder Integer Kontext-Identifizierung in der Help-Datei </div> </div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;"> logischer_window_name liefert nichts </div> <div style="text-align: center;"> Zeiger laut open() </div> </div>
<code>.showModalDialog()</code>	Modales Dialog-Fenster erzeugen und anzeigen (modaler Dialog) sowie automatisch focussieren Syntax: <div style="text-align: center;"> <code>[var Wert =] logischer_window_name.showModalDialog(URL <div style="display: flex; justify-content: center; margin-top: 5px;"> [, Arguments] [, Features] </div>) </code> </div>

URL	String als Url des zu ladendenen HTML-Dokumentes
-----	--



Arguments	freie Parameterliste für Werteübergabe Dialog-Fenster Werteübergabe auch per Felder möglich Format wie Eigenschaft .dialogArguments wenn Liste als String, so max 4096 Zeichen in der Liste
Features	String als Parameterliste mit Kommatrennung gesamte Liste in " " setzen bzw. ' ' z.B. "center:yes, status:yes" Listenelement: option:wert dialogHeight:hoehe_in_pixel ab IE 5.x auch Gleitkomma cm, mm, in, pt, pc, oder px mindestens 100 Pixel (wenn < 100 so automatisch 100 verwendet) dialogLeft:X_koordinate_relativ_linke_obere_Ecke_Screen (0,0) dialogTop:Y_koordinate_relativ_linke_obere_Ecke_Screen (0,0) dialogWidth:breite_in_pixel ab IE 5.x auch Gleitkomma cm, mm, in, pt, pc, oder px center:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Dialogbox im Screen zentrieren dialogHide:{ yes no 1 0 on off } default ist no bzw. 0 yes bzw. 1 Dialogbox nicht druckbar bzw. nicht angezeigt in Druckvorschau edge:{ sunken raised } default ist raised Kantenstil sunken = versenkt raised = nicht versenkt help:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Anzeige des Fragezeichen in der Titelzeile (context-sensitive Help icon) resizable:{ yes no 1 0 on off } default ist no bzw. 0 yes bzw. 1 Dialogboxgröße veränderbar scroll:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Inhalt scrollbar status:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Statuszeile anzeigen unadorned:{ yes no 1 0 on off } default ist no bzw. 0 yes bzw. 1 Rahmen farbig anzeigen

Wert im Format laut Eigenschaft .returnValue

logischer_window_name Zeiger laut open()

Beispiel 1:

```
<SCRIPT>
function ErmittleZufallsZahl(Faktor)
{return parseInt(Math.random() * Faktor);}

function BoxHoeheErmittleIn()
{
    var OptionsFeld = ID_Formular.ID_Select.options;
    var OptionsFeldIndex = ID_Formular.ID_Select.selectedIndex;
    var OptionsWert = OptionsFeld [OptionsFeldIndex].text;

    // auf Auswahl " Zufallshoehe" prüfen
    if (OptionsWert.indexOf("Zufallshoehe") > -1 )
    {OptionsWert = ErmittleZufallsZahl(document.body.clientHeight);}

    // Features für Boxöffnen ermitteln
```



```

        var BoxHoeheFeatures="dialogHeight:" + OptionsWert + "px;";

        // und liefern
        return BoxHoeheFeatures;
    }

    function BoxOeffnen()
    {
        var BoxHoeheFeatures= BoxHoeheErmitteln();

        window.showModalDialog( "test.htm",
                                "",
                                BoxHoeheFeatures
                                );
    }
</SCRIPT>
<FORM NAME="ID_Formular">
    wachle Boxhoehe in Pixel
    <SELECT NAME="ID_Select">
        <OPTION>Zufallshoehe
        <OPTION>150
        <OPTION>200
        <OPTION>250
        <OPTION>300
    </SELECT>
    oeffne Modal Dialog Box
    <INPUT TYPE="button" VALUE="Box oeffnen" onclick="BoxOeffnen()">
</FORM>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeige()
    {
        // Zeiger auf die Formularelemente holen
        var FormularElemente = ID_Formular.elements;

        // Formulardaten in einem privaten Objekt kapseln als Argument für modalen Dialog
        var FormularDaten = new Object();
        FormularDaten.firstName = FormularElemente.ID_Input1.value;
        FormularDaten.lastName = FormularElemente.ID_Input2.value;

        // Fenster als modalen Dialog anzeigen und dabei Eigenschaft .dialogArguments füllen
        // Mit der Übergabe der Daten erfolgt das Öffnen des neuen Fenster, das
        // sich auf die namensgleichen Eigenschaften von FormularDaten
        // beruft, deren Bezeichner mit in den Argumenten übergeben werden
        window.showModalDialog( "test.htm",
                                FormularDaten,
                                "dialogHeight:300px; dialogLeft:200px;"
                                );
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM ID= "ID_Formular">
        Vorname:
        <INPUT TYPE="text" NAME="ID_Input1" VALUE="Vorname">
        <BR>
        Nachname:
        <INPUT TYPE="text" NAME="ID_Input2" VALUE="Nachname">
    </FORM>
    <BR>
    <BUTTON onclick="Anzeige();" >Formulardaten im modalen Dialog anzeigen</BUTTON>
</BODY>
</HTML>

```

test.htm enthält:

```

<HTML>
<HEAD>

```



```

<SCRIPT>
    function Anzeigen()
    {
        var DialogArgumente = window.dialogArguments;

        // Es müssen namensidentische Bezeichner der Eigenschaften von
        // DialogArgumente verwendet werden:
        //     firstName und lastName werden in der
        //     aufrufenden Webseite definiert
        //     und müssen hier ebenfalls verwendet werden
        document.writeln("Vorname = " + DialogArgumente.firstName);
        document.write("Nachname = " + DialogArgumente.lastName);
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeigen();">
</BODY>
</HTML>

```

.showModelessDialog()

nicht-modales Fenster erzeugen, aber anzeigen nur, wenn Focus geändert wird
 verwendbar für Desing von Menüs
 Tooltips

Syntax:

```

[ var Wert = ] logischer_window_name.showModelessDialog(URL
                                                         [, Arguments]
                                                         [, Features]
                                                         )

```

URL	String als Url des zu ladendenen HTML-Dokumentes
Arguments	freie Parameterliste für Werteübergabe an Dialogbox Werteübergabe auch per Felder möglich Format wie Eigenschaft .dialogArguments wenn Liste als String, so max 4096 Zeichen in der Liste
Features	String als Parameterliste mit Kommatrennung gesamte Liste in " " setzen bzw. ' ' z.B. "center:yes; status:yes" Listenelement: option:wert dialogHeight:hoehe_in_pixel ab IE 5.x auch Gleitkomma cm, mm, in, pt, pc, oder px mindestens 100 Pixel (wenn < 100 so automatisch 100 verwendet) dialogLeft:X_koordinate_relativ_linke_obere_Ecke_Screen (0,0) dialogTop:Y_koordinate_relativ_linke_obere_Ecke_Screen (0,0) dialogWidth:breite_in_pixel ab IE 5.x auch Gleitkomma cm, mm, in, pt, pc, oder px center:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Dialogbox im Screen zentrieren dialogHide:{ yes no 1 0 on off } default ist no bzw. 0 yes bzw. 1 Dialogbox nicht druckbar bzw. nicht angezeigt in Druckvorschau edge:{ sunken raised } default ist raised Kantenstil sunken = versenkt raised = nicht versenkt help:{ yes no 1 0 on off } default ist yes bzw. 1 yes bzw. 1 Anzeige des Fragezeichen in der Titelzeile (context-sensitive Help icon) resizable:{ yes no 1 0 on off } default ist no bzw. 0 yes bzw. 1 Dialogboxgröße veränderbar scroll:{ yes no 1 0 on off }




```

default ist yes bzw. 1
yes bzw. 1      Inhalt scrollbar
status:{ yes | no | 1 | 0 | on | off }
default ist yes bzw. 1
yes bzw. 1 Statuszeile anzeigen
unadorned:{ yes | no | 1 | 0 | on | off }
default ist no bzw. 0
yes bzw. 1 Rahmen farbig anzeigen

```

Wert im Format laut Eigenschaft .returnValue

logischer_window_name Zeiger laut open()

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    var Vorname="";      // muss global sein da in Test.htm benutzt

    function VornameAnzeigen()
    {
        showModelessDialog( "Test.htm",      // Url der Dialog-Box
                             window,
                             "status:false;dialogWidth:300px;dialogHeight:300px"
                           );
    }

    function VornameAktualisieren()      // Funktion wird in Test.htm aufgerufen
    { ID_Span.innerText = Vorname;}

</SCRIPT>
</HEAD>
<BODY>
    <P> aktueller Vorname ist :
        <SPAN ID="ID_Span"
            STYLE="color:red;font-size:24">
            unbekannt
        </SPAN>
    </P>
    <INPUT TYPE="button"
        VALUE="öffnen der Modeless Dialog Box"
        onclick="VornameAnzeigen()">

</BODY>
</HTML>

```

Kode für *Test.htm*, also dem Inhalt der Box

```

<HTML>
<HEAD>
<TITLE>Test.htm</TITLE>
<SCRIPT>
    function VornameAktuellAnzeigen()
    {
        var DialogArgumente = dialogArguments;
        DialogArgumente.Vorname = ID_Input.value;
        DialogArgumente.VornameAktualisieren();
        // Aufruf einer Funktion aus
        // Elterndokument
    }

    function Init()
    {
        var DialogArgumente = dialogArguments;
        DialogArgumente.Vorname = "unbekannt";
        DialogArgumente.VornameAktualisieren();
        // Aufruf einer Funktion aus
        // Elterndokument
    }

</SCRIPT>
</HEAD>
<BODY>
    <LABEL FOR="ID_Input" ACCESSKEY="v">
        Gib den

```



```

        <SPAN STYLE="text-decoration:underline">
        V
        </SPAN>ornamen ein :
    </LABEL>
    <INPUT ID= "ID_Input">
    <BR><BR>
    <INPUT VALUE="Anzeige aktueller Vorname und Box offen lassen"
        TYPE=button
        onclick=" VornameAktuellAnzeigen();">

    <INPUT VALUE=" Anzeige aktueller Vorname und Box schliessen"
        TYPE=button
        onclick=" VornameAktuellAnzeigen();window.close();">

    <INPUT VALUE="Init"
        TYPE=button
        onclick=" Init();window.close();"
    >
</BODY>
</HTML>

```

4.3.2.2.1. window. clientInformation Objekt des Internet Explorer

ab IE 4.x

beinhaltet die Informationen über den Browser (Client)

Informationen über den Browser (Client) werden microsoft-spezifisch zusätzlich im Behavior clientCaps verwaltet

z.B. per Active-X installierbare Komponenten des IE
der Online-Zustand des Users zum Netzwerk
die Microsoft Virtuelle Maschine für Java

Erzeugung:

durch Browser

Zugriff:

window.clientInformation.eigenschaft	oder	clientInformation.eigenschaft
window.clientInformation.methode()	oder	clientInformation.methode()

logischer_window_name.clientInformation.eigenschaft
logischer_window_name.clientInformation.methode()

logischer_window_name Zeiger laut open()

Eigenschaften:

.appName

Codename des Browsers

Syntax:

[var Kette =] logischer_window_name.clientInformation.appCodeName

Kette

String

Default ist "Mozilla"

logischer_window_name

Zeiger laut open()

nur lesen

.appMinorVersion

Browser-Versionsnummer: Unternummer der Hauptnummer

z.B. 1 des IE 4.1

Syntax:

[var Wert =] logischer_window_name.clientInformation.appMinorVersion

Wert

Integer

Achtung Opera-Browser gibt sich als IE aus !

logischer_window_name

Zeiger laut open()

nur lesen

.appName

Browsersname

Syntax:

[var Kette =] logischer_window_name.clientInformation.appName

Kette

String

Default ist "Microsoft Internet Explorer"

"Netscape" ist der Netscape Navigator.

Achtung Opera-Browser gibt sich als IE aus !

logischer_window_name

Zeiger laut open()

.appVersion

Plattform und Browserversion



Aufbau

clientVersion (platform; information; extraInformation)

oder 5.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)Hinweis: Haupt-Versionsnummer ermittelbar per .parseFloat(navigator.appVersion)
z.B. 4 des IE 4.1

Syntax:

[var Kette =] logischer_window_name.clientInformation.appVersion

Kette String im Format

clientVersion (platform; information; extraInformation)

z.B. "5.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)"

logischer_window_name

Zeiger laut open()

nur lesen

.browserLanguage

Browsersprache

Achtung: ab IE 5.x Sprache **immer** laut **regionalen** Einstellungen des Users im **Betriebssystem**

Syntax:

[var Kette =] logischer_window_name.clientInformation.browserLanguage

Kette

String

unter 5.x: Sprache des Browsers

ab 5.x: Sprache des Betriebssystems

logischer_window_name

Zeiger laut open()

nur lesen

.cookieEnabled

Cookienutzbarkeit im Browser

Syntax:

[var Wert =] logischer_window_name.clientInformation.cookieEnabled

Wert

false

Browser unterstützt keine Cookies

true

Browser unterstützt Cookies

logischer_window_name

Zeiger laut open()

nur lesen

.cpuClass

CPU-Hersteller

Syntax:

[var Kette =] logischer_window_name.clientInformation.cpuClass

Kette

String

"x86"

Intel

"68K"

Motorola

"Alpha"

Digital

"PPC"

Motorola

"Other"

alles andere inklusive Sun SPARC.

logischer_window_name

Zeiger laut open()

nur lesen

.onLine

System-Online-Status (nicht Netzwerk-Online-Status !!!)

Syntax:

[var Wert =] logischer_window_name.clientInformation.onLine

Wert

true

System ist online

false

System ist offline

logischer_window_name

Zeiger laut open()

nur lesen

Hinweis: Eigenschaft .connectionType des Behavior clientCaps zeigt den Status der
genutzten Verbindung an

.platform

Betriebssystem

Syntax:

[var Kette =] logischer_window_name.clientInformation.platform

Kette

String



		"HP-UX"	HP Unix	
		"MacPPC"		Macintosh Power-
	PC			
		"Mac68K"		Macintosh 68K-
	based			Computer
		"SunOS"	Solaris	
		"Win32"	Windows 32-Bit	
		"Win16"	Windows 16-Bit	
		"WinCE"	Windows CE	
	logischer_window_name	Zeiger laut open()		
nur lesen				
.systemLanguage	Standardsprache des Betriebssystems			
	Syntax:			
	[var Kette =] logischer_window_name.clientInformation.systemLanguage			
	Kette	String		
		Sprachcode		
		z.B. "en" oder "de"		
	logischer_window_name	Zeiger laut open()		
nur lesen				
.userAgent	HTTP User-Agent			
	Syntax:			
	[var Wert =] logischer_window_name.clientInformation.userAgent			
	Wert	String		
		z.B. IE 6.0 liefert unter Windows XP		
		"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"		
	logischer_window_name	Zeiger laut open()		
nur lesen				
.userLanguage	Sprache des Betriebssystems laut UserEinstellung			
	Syntax:			
	[var Kette =] logischer_window_name.clientInformation.userLanguage			
	Kette	String		
		Sprachcode		
		z.B. "en" oder "de"		
	logischer_window_name	Zeiger laut open()		
nur lesen				
Methoden:				
.javaEnabled()	Java-Verfügbarkeit			
	ab IE 6.x			
	siehe Objekt window.clientInformation			
	Syntax:			
	[var Wert =] logischer_window_name.clientInformation.javaEnabled()			
	Wert	true	Java ist nutzbar	
		false	Java ist nicht nutzbar	
	logischer_window_name	Zeiger laut open()		
Hinweis: Eigenschaft .javaEnabled des Behavior clientCaps zeigt die Verfügbarkeit der				
Microsoft Virtuellen Maschine für Java an				
Achtung: Aufgrund eines Streites mit dem Unternehmen Sun entwickelt Microsoft ab IE 6.x die MS VM				
nicht mehr weiter, hält die VM aber noch per Download verfügbar. Für eine aktuellere Version				
muss der User selbständig beim Unternehmen Sun eine Lizenz für Java erwerben und die				
zugehörige Software installieren (inklusive Updates). Es reicht dabei die Run-Time-Version der				
Virtuellen Java Maschine von Sun. Ob allerdings Sun-Java-Standards in Microsoft-Produkten				
implementiert werden, ist nicht gesichert.				
.taintEnabled()	Data Tainting-Verfügbarkeit			
	siehe Objekt window.clientInformation			
	Syntax:			



```
[ var Wert = ] logischer_window_name.clientInformation.taintEnabled()
```

Wert	true	Data tainting ist verfügbar ab IE 6.x
	false	Data tainting is nicht verfügbar immer geliefert unter IE 6.x

```
logischer_window_name      Zeiger laut open()
```

4.3.2.2.2. window.clipboardData Objekt des Internet Explorer

Unterstützung des Zugriffs auf vordefinierte Formate für Clipboard-Nutzung

Es wird das System-Clipboard verwendet. Mehrfacheinfügung ist möglich.

ab IE 5.x

siehe auch Objekt event.dataTransfer des IE

Erzeugung:

durch Browser

Zugriff:

```
window.clipboardData.methode()                      oder      clipboardData.methode()
```

```
logischer_window_name.clipboardData.methode()
```

```
logischer_window_name      Zeiger laut open()
```

Beispiel für Clipboard-Nutzung ohne Drag&Drop:

```
<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var TextBereich = document.body.createTextRange();
        TextBereich.findText(ID_Div1.innerHTML);
        TextBereich.select(); // selektieren
    }

    function VorDemAusschneiden()
    {event.returnValue = false;} // Cut per Menü aktivieren

    function Ausschneiden()
    {
        ID_Div1.innerHTML = "";
        ID_Input.innerHTML += window.clipboardData.setData("Text", ID_Div1.innerHTML);
        // true oder false
        // Clipboard-Daten sind Texttyp
        event.returnValue = false;
    }

    function VorDemEinfuegen()
    {event.returnValue = false;} // Paste per Menü aktivieren

    function Einfuegen()
    {
        ID_Div2.innerHTML = window.clipboardData.getData("Text");
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID=" ID_Div1" onbeforecut="VorDemAusschneiden()" oncut="Ausschneiden()">
        Dieses Text selektieren und ausschneiden
    </DIV>
    <DIV ID="ID_Div2" onbeforepaste="VorDemEinfuegen()" onpaste="Einfuegen()">
        den ausgeschnittenen Text hier einfügen
    </DIV><BR>
    <INPUT ID="ID_Input" TYPE="text" READONLY VALUE="" SIZE="6">
</BODY>
</HTML>
```

Beispiele für Clipboard-Nutzung mit Drag&Drop:

```
<HEAD>
<SCRIPT>
```



```

function DragStart() // Url eines Ankers als Datenformat festlegen und Daten holen
{event.dataTransfer.setData("URL", ID_A.href);}

function DragEnde() // Daten im URL-Format als Textdaten in den Span ablegen
{ID_Span.innerText = event.dataTransfer.getData("URL");}
</SCRIPT>
</HEAD>
<BODY>
  <A ID="ID_A" HREF="anker"
    onclick="return(false);"
    ondragstart=" DragStart()"
  >
    Testanker
  </A>
  <SPAN ID="ID_Span" ondragenter=" DragEnde()">
    obigen Link auf diese Stelle hierher dropfen
  </SPAN>
</BODY>

```

```

<HEAD>
<SCRIPT>
  function InitiateDrag()
  {event.dataTransfer.setData("URL", ID_Image.src);}

  function FinishDrag()
  {ID_Span.innerText = event.dataTransfer.getData("URL");}
</SCRIPT>
</HEAD>
<BODY>
  <IMAGE ID="ID_Image" SRC="/test/graphics/test.gif"
    ondragstart="InitiateDrag()">
  <SPAN ID="ID_Span" ondragenter="FinishDrag()"
  >
    Image hierher dropfen
  </SPAN>
</BODY>

```

```

<HTML>
<HEAD>
<SCRIPT>
  var Wert;

  function TextBereichErzeugen()
  {
    // Text im gesamten Body adressieren
    var TextBereich = document.body.createTextRange();

    // davon den Textteil des Source-Div adressieren
    TextBereich.findText(ID_DivSource.innerText);

    // und diesen selektieren
    TextBereich.select();
  }

  // Ausschneiden aktivieren, da standardgemäß für DIV deaktiv ist
  // Ausschneiden bedeutet: Browser verschiebt automatisch in das Clipboard
  function AusschneidenAktivieren() // ist Eventhandler für onbeforecut
  {event.returnValue = false;} // Event-Handler-Rückkehrcode

  function Ausschneiden() // ist Eventhandler für oncut
  {
    // Clipboard füllen mit Daten vom Texttyp
    // als Text des Source-Div
    Wert = window.clipboardData.setData("Text", ID_DivSource.innerText);

    // Source-Div Textbereich löschen
    ID_DivSource.innerText = "";
  }

```



```

// Rückgabewert vom Clipboardfüllen als Text (true oder false)
//      im Text des Input-Button anzeigen
ID_Input.innerText += Wert; // Wert mit als Text

// Event-Handler-Rückkehrcode
event.returnValue = false;
}
// Einfügen aktivieren, da standardgemäß für DIV deaktiv ist
// Einfügen bedeutet: Browser fügt aus Clipboard in das Zielobjekt ein
function EinfuegenAktivieren() // ist Eventhandler für onbeforepaste
{ event.returnValue = false; } // Event-Handler-Rückkehrcode

function Einfuegen() // ist Eventhandler für onpaste
{
    // aus Clipboard Daten vom Texttyp holen und in den Textbereich
    // des Target-Div ablegen
    ID_DivTarget.innerText = window.clipboardData.getData("Text");

    // Event-Handler-Rückkehrcode
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY onload="TextBereichErzeugen()">
  <DIV ID="ID_DivSource" onbeforecut="AusschneidenAktivieren()"
    oncut="Ausschneiden()"
  >
    Diesen Text mit Maus markieren und ausschneiden
  </DIV>
  <DIV ID="ID_DivTarget" onbeforepaste="EinfuegenAktivieren()"
    onpaste="Einfuegen()"
  >
    An diese Stelle den ausgeschnittenen Text einfügen
  </DIV>
  <BR>
  <INPUT ID="ID_Input" TYPE="text" VALUE="Rückkehrcode = ">
</BODY>
</HTML>

```

```

<HTML>
<HEAD>
<SCRIPT>
// ##### Quellobjekt #####
function EventHandlerFuerOnDragStart()
    // Quellobjekt löst Event aus:
    //      in Quelle wird markiert und selektiert
    {
        // selektierte Quell-Daten in die Zwischenablage puffern
        //      (Puffer wird per window.event-Objekt verwaltet)
        //      Datentyp ist hier uninteressant, da für die Zwischenablage
        //      der Typ automatisch erkannt wird, also
        //      Speicherung der Daten in der Zwischenablage
        //      immer typgerecht ist
        var ZwischenAblage = window.event.dataTransfer;

        // und diese Daten als verschiebbar erklären:
        //      aus der Quelle in die Zwischenablage
        ZwischenAblage.effectAllowed = "move";
    }

// ##### Zielobjekt #####
function EventHandlerFuerOnDragEnter
    // Zielobjekt löst Event aus
    // Maus über Ziel nach dem Draggen der Daten,
    //      UND Maustaste ist noch nicht losgelassen
    // also Zielobjekt wird mit den Daten betreten
    {

```



```

        // Daten adressieren
        var ZwischenAblage = window.event.dataTransfer;

        // und diese Daten als verschiebbar erklären:
        //      aus der Zwischenablage in das Zielobjekt
        ZwischenAblage.dropEffect = "move";

        // Event-Handler-Rückkehrcode
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }

    function EventHandlerFuerOnDrop
        // Zielobjekt löst Event aus
        // Maus über Ziel nach dem Dropfen der Daten,
        //      UND Maustaste wird losgelassen
    {

        // Ziel adressieren, das mit ondrop-Ereignis behandelt werden soll
        var Ziel = window.event.srcElement;

        // Daten in der Zwischenablage adressieren und holen
        //      (Puffer wird per window.event-Objekt verwaltet)
        var ZwischenAblage = window.event.dataTransfer;

        // und Daten vom Texttyp aus der Zwischenablage im Ziel ablegen,
        //      da Ziel nur Textdaten empfangen kann
        Ziel.innerText += Daten.getData("text");

        // Event-Handler-Rückkehrcode
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }

    function EventHandlerFuerOnDragOver
        // Zielobjekt löst Event aus
    {

        // nichts tun ausser Rückkehrcode des Handlers liefern
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ondragstart=" EventHandlerFuerOnDragStart ()"
    >
        Diesen Text markieren und draggen
    </DIV>
    <BR>
    <DIV
        ondragover="EventHandlerFuerOnDragOver()"
        ondragenter="EventHandlerFuerOnDragEnter()"
        ondrop="EventHandlerFuerOnDrop()"
    >
        An diese Stelle den Text dropfen
    </DIV>
</BODY>
</HTML>

```

Eigenschaften:

keine

Methoden:

.clearData()	Clipboardinhalt löschen
.getData()	Anwendung für Ereignisse ondragstart oder ondrop Clipboard auslesen
.setData()	Anwendung für Ereignisse oncopy oder oncut Handler muss liefern window.event.returnValue=false; Clipboard füllen, also Daten dort ablegen wenn Clipboard nicht leer so immer anhängen

4.3.2.2.3. window.dialogArguments Objekt des Internet Explorer

Dieses Objekt verwaltet die Argumentenübergabe für ein modales oder nicht modales Dialogfenster.
Das Objekt kann folgende Datentypen der Argumente verwalten:

Stringvariable



numerische Variable
Objekt
Feld
Function

Sämtliche Argumente sind im Dialogfenster nur lesbar.

ab IE 5.x

Erzeugung:

Ein Dokument kann den Dialog mit folgenden Methoden erzeugen:

window.showModalDialog() oder .showModalDialog()
bzw. window.showModelessDialog() oder .showModelessDialog()

logischer_window_name.showModalDialog()
bzw. logischer_window_name.showModelessDialog()

logischer_window_name Zeiger laut open()

Diese Methoden erzeugen zum Elternfenster, dass die Methoden aufruft, ein neues Fenster für Dialog (Kind-Fenster).

Zugriff im Kindfenster:

var Zeiger = window.dialogArguments

Zeiger.eigenschaft
Zeiger.methode()

Zeiger auf das Objekt
eigenschaft und methode() laut Elternfenster (Elterndokument) also **namensindentliche Referenz**

Beispiel:

Dokument, das den Dialog erzeugt (Elterndokument):

```
<HTML>
<HEAD>
<SCRIPT>
    function Anzeige()
    {
        // Zeiger auf die Formularelemente holen
        var FormularElemente = ID_Formular.elements;

        // Formulardaten in einem privaten Objekt kapseln als Argument für modalen Dialog
        var FormularDaten = new Object();
        FormularDaten.firstName = FormularElemente.ID_Input1.value;
        FormularDaten.lastName = FormularElemente.ID_Input2.value;

        // Fenster als modalen Dialog anzeigen und dabei Eigenschaft .dialogArguments füllen
        // Mit der Übergabe der Daten erfolgt das Öffnen des neuen Fenster, das
        // sich auf die namensgleichen Eigenschaften von FormularDaten
        // bezieht, deren Bezeichner mit in den Argumenten übergeben werden
        window.showModalDialog("test.htm",
                               FormularDaten,
                               "dialogHeight:300px; dialogLeft:200px;"
                               );
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM ID= "ID_Formular">
        Vorname:
        <INPUT TYPE="text" NAME="ID_Input1" VALUE="Vorname">
        <BR>
        Nachname:
        <INPUT TYPE="text" NAME="ID_Input2" VALUE="Nachname">
    </FORM>
    <BR>
    <BUTTON onclick="Anzeige();" >Formulardaten im modalen Dialog anzeigen</BUTTON>
</BODY>
</HTML>
```

Dokument test.htm das den HTML-Container des Dialoges darstellt (Kind-Dokument):

```
<HTML>
<HEAD>
```



```

<SCRIPT>
    function Anzeigen()
    {
        var DialogArgumente = window.dialogArguments;

        // Es müssen namensidentische Bezeichner der Eigenschaften von
        //     DialogArgumente verwendet werden:
        //         firstName und lastName werden in der
        //         aufrufenden Webseite definiert
        //         und müssen hier ebenfalls verwendet werden
        document.writeln("Vorname = " + DialogArgumente.firstName);
        document.write("Nachname = " + DialogArgumente.lastName);
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeigen();">
</BODY>
</HTML>

```

4.3.2.2.4. window.document Objekt des Internet Explorer

Achtung: Das Objekt document.body ist nicht mehr verfügbar und wurde durch das Objekt document.documentElement ersetzt !

document.documentElement ist der Zeiger auf den Body und nicht mehr document.body !

Beispiel zur Abfrage auf CSS1-Konformität und damit zur Verwendung von document.body bzw. document.documentElement

```

function DocTypeAbfrage()
{
    // Annahme: CSS1-Standard, also !DOCTYPE wurde kodiert
    var ZeigerAufDokument=document.documentElement;

    if (document.compatMode=="BackCompat")
    {
        // nicht CSS1-Standard, also wurde kein !DOCTYPE kodiert
        ZeigerAufDokument =document.body;
    }

    return ZeigerAufDokument;
}

```

Hinweise: document.compatMode ist nur lesbar, also !DOCTYPE nicht setzbar

Wert "CSS1Compat" ab IE 6
geliefert wenn !DOCTYPE kodiert wurde
Rendern mit CSS1 Standard (Standard-Compliant-Modus)

Wert "BackCompat" ab IE 3
Geliefert wenn kein !DOCTYPE kodiert wurde
Rendern mit Nicht-kein Standard-Compliant-Modus

Ansatz:

Das Objekt ist das HTML-Dokument, das in Browserfenster geladen und angezeigt wird.
ist der Container für die HTML-Elemente der Webseite z.B. BODY oder DIV
umfasst sämtlichen Code **zwischen** <HTML...> und </HTML>:
Der Code hinter </HTML> (z.B. Script) wird nicht geparkt, aber eventuell als Plain-Text angezeigt.

Scriptsteuerung erst ab IE 4.x

Hinweis: Es gibt noch das html Objekt, welches den **HTML-Tag** beschreibt.

Das HTML-Dokument im HTML-DOM:

Das Objekt wird als Hierarchie von Knoten dargestellt, die nicht der Folge der Elemente im HTML-Code entspricht. Ein HTML-Element ist ein Knoten als Kind des Dokumentes.

Das HTML-Element im HTML-DOM:

Das HTML-Element wird in seine Komponenten zerlegt, die Knoten und Kinder des HTML-Elementes sind.
Bsp.: SPAN-Element mit Text: Text ist ein Knoten von SPAN

Zur Referenzierung von HTML-Elementen des Dokumentes sollte im Regelfall **document.all** kodiert werden, damit die Browserperformance steigt (siehe Collection document.all).

Das HTML-Element beim IE und NS:

als HTML-Tag: Der IE und NS unterstützen z.T. verschiedene Tags.



als Objekt: Es gibt Objekte, die im IE und NS implementiert sind aber z.T. auf verschiedene Weise
z.B. Objekt document

DOM und Collectionen zum Objekt document:

Zur Verwaltung des Dokumentes existieren Collectionen als Felder, die zugleich als Schnittstelle zum Programmierer dienen. Collectionen des Objektes document werden immer ohne .all kodiert, da sie keine HTML-Elemente darstellen. Der Netscape besitzt teilweise Collectionen, die auch der Internet Explorer bedient.

Objekt document und Browserfenster:

Das HTML-Dokument wird getrennt vom Fensterobjekt verwaltet: Ein Dokument muss zwar in ein Fenster geladen werden, aber das Dokument wird nur dann visualisiert (gerendert), wenn es sichtbare Elemente besitzt. Die Referenzierung des Dokumentes **im** Fenster erfolgt anhand des

logischen Fenstername, der geliefert wird

per Methode .open() zum window Objekt und ein frei festgelegter oder vordefinierter Name sein kann als Wert des ID-Attributes bei FRAMESET mit FRAMES bzw. IFRAMES (siehe dort).

HTML-Dokument und Druck-Layout im Browserfenster:

Im Internet Explorer ab 5.x lässt sich das Druck-Layout der Seite, also des Dokumentes, bezüglich Kopf- und Fusszeile beeinflussen:

Menüpunkt Datei-Seite einrichten und dort die Angaben editieren.

Die Angaben haben folgende vordefinierte Komponenten:

&w	Platzhalter für den Titel des Fensters, in dem das Dokument angezeigt wird (siehe auch Objekt window)
&u	Platzhalter für die URL des Dokumentes (siehe auch Objekt location)
&d	Platzhalter für das Datum im Kurzformat laut Uhr des PC des Users
&D	Platzhalter für das Datum im Langformat laut Uhr des PC des Users
&t	Platzhalter für die Uhrzeit im 12-Stunden-Format laut Uhr des PC des Users
&T	Platzhalter für die Uhrzeit im 24-Stunden-Format laut Uhr des PC des Users
&p	Platzhalter für die Seitennummer
&P	Platzhalter für die Gesamtanzahl der Seiten im Dokument
&b	Text zentrieren, der hinter &b kodiert wird
&b&b	Text rechtsbündig setzen, der hinter &b&b kodiert wird
&&	das Zeichen "&"

Das Layout ist in der Seitenansicht sichtbar, wobei die Platzhalter durch konkrete Werte ersetzt sind (als wenn gedruckt werden würde).

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE:

Folgende Reihenfolge wird beim Parsen eingehalten:

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !
Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu den Beschreibungen der Objekte und Collectionen aus der Hierarchie des Dokumentes:

Die Beschreibungen beziehen sich in der Regel aus Gründen der Vereinfachung auf das aktuelle Fenster und dessen (aktuelles) Dokument. sind analog anwendbar für ein per logischem Windownamen adressiertes Dokument.

Erzeugung:

durch den Browser oder per Methode .createDocumentFragment()

Beispiel für Ausdruck des aktuellen Dokumentes:

```
<BODY>
  <INPUT TYPE="button" VALUE="Drucken" onclick="javascript:self.print()">
</BODY>
```

Beispiel für Webseite mit eigenem Icon ab IE 5.x:

```
<LINK REL="SHORTCUT ICON" HREF="name.ico">
```



name.ico: name ist beliebig, auch mit Pfad

ICO-Datei: 32x32 Pixel mit 16 Farben
 oder 16x16 Pixel mit 256 Farben
 ist BMP-Datei, die zu ICO-Datei konvertiert werden muss
 (kann nicht jedes Grafik-Programm, aber Microsoft bietet dafür IconPro an)

Beispiel für Ermittlung der Verweilzeit des Users auf der Webseite:

```
<HTML>
<HEAD>
<SCRIPT>
<!--
    var start_zeit;      // muss global sein

    function start()
    { start_zeit = new Date(); }

    function ende()
    {
        var ende_zeit = new Date();
        var wert=ende_zeit.getTime() - start_zeit.getTime();
        wert = wert /1000;    // in Sekunden umrechnen
        alert("Verweilzeit in Sekunden: " + Math.floor(wert).toString());
    }
//-->
</SCRIPT>
</HEAD>

<BODY ... onLoad="start();" onUnload="ende();">
</BODY>
</HTML>
```

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x):

Seitenteile, die nicht gedruckt werden sollen, in <DIV CLASS="keindruck"> und </DIV>
 oder in und einschliessen.

zwischen <HEAD> und </HEAD> einfügen: <LINK REL="stylesheet" MEDIA="print" HREF="print.css">
 print.css enthält nur .keindruck {display:none;}

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

Seitenteile, die nur auf Ausdrucken sichtbar sein sollen, in <DIV CLASS="nurdruck"> und </DIV>
 oder in und einschliessen

zwischen <HEAD> und </HEAD> einfügen: <LINK REL="stylesheet" MEDIA="screen" HREF="screen.css">
 screen.css enthält nur .nurdruck {display:none;}

Beispiel für Dokumenten-Hintergrund ein- bzw. ausblenden:

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
//      Dokument-Hintergrundfarbe ein- und ausblenden
//      einblenden beim Dokument laden
//      ausblenden beim Wechsel zu anderem Dokument, also entladen des aktuellen Dokumentes
//      mit Farbveränderung rückwärts gegenüber Laden

//*****
//
//      Nachfolgende Variablen muss der Programmierer anpassen
//
//*****
//      Farbe wird aus Rot- und Gruen- und Blauanteil gebildet
//      Alle Farbanteilwerte gelten von 0 bis 255, alles ganzzahlig, wird nicht geprüft
//      Startwerte für Startfarbe des Ein- bzw. Ausblenden, müssen <= Endwerte sein, wird nicht geprüft
//      Endwerte für Endfarbe nach Ein- bzw. Ausblenden, müssen >= Startwerte sein, wird nicht geprüft
```



```

//      Farbschritte = Stufen für Ein- bzw. Ausblenden
//      je mehr um so langsamer das Ein- bzw. Ausblenden
//      nur ganzzahlig und maximal die kleinste paarige Differenz der obigen Werte
//      (keine Prüfung)

var RotAnteil_Start      = 0;
var GruenAnteil_Start    = 0;
var BlauAnteil_Start     = 0;
var RotAnteil_Ende       = 255;
var GruenAnteil_Ende     = 255;
var BlauAnteil_Ende      = 255;
var FarbSchritte         = 64;

// *****
//
//      Nachfolgenden Code nicht verändern
//
// *****

function DezimalZuHexaString(DezimalerWert)
// liefert String von 2 Ziffern bzw. Grossbuchstaben
// X_DezimalerWert  Gleitkomma
//
//      wenn nicht ganzzahlig, so vor der Konvertierung ganzzahlig gemacht
//      wenn <= 0   so '00' geliefert
//      wenn >= 255 so 'FF' geliefert
{
    var HexaZeichenFeld = "0123456789ABCDEF";
    var HexaKette="";

    // Dezimalwert zu ganzzahlig
    X_DezimalerWert=Math.floor(X_DezimalerWert);

    if (DezimalerWert < 0)
    {HexaKette="00";}
    else
    {
        if (DezimalerWert > 255)
        {HexaKette="FF";}
        else
        {
            //      Indexe im HexaZeichenFeld ermitteln
            var HexaKette_Zeichen1_PositionImHexaZeichenFeld = Math.floor(DezimalerWert / 16);
            var HexaKette_Zeichen2_PositionImHexaZeichenFeld =
                DezimalerWert - (HexaKette_Zeichen1_PositionImHexaZeichenFeld * 16);

            //      HexaKette bilden
            HexaKette =
                HexaZeichenFeld.charAt(HexaKette_Zeichen1_PositionImHexaZeichenFeld)
                + HexaZeichenFeld.charAt(HexaKette_Zeichen2_PositionImHexaZeichenFeld);
        }
    }

    return HexaKette;
}

function BackgroundEinAusBlenden(RotAnteil_Start,GruenAnteil_Start,BlauAnteil_Start,
    RotAnteil_Ende,GruenAnteil_Ende,BlauAnteil_Ende,
    FarbSchritte
)
{
    //      Farbe wird aus Rot- und Gruen- und Blauanteil gebildet
    //      Alle Farbanteilwerte gelten von 0 bis 255, alles ganzzahlig, wird nicht geprüft
    //      Startwerte für Startfarbe des Ein- bzw. Ausblenden, müssen <= Endwerte sein, wird nicht geprüft
    //      Endwerte für Endfarbe nach Ein- bzw. Ausblenden, müssen >= Startwerte sein, wird nicht geprüft
    //      Farbschritte = Stufen für Ein- bzw. Ausblenden
    //      je mehr um so langsamer das Ein- bzw. Ausblenden
    //      nur ganzzahlig und maximal die kleinste paarige Differenz der obigen Werte
    //      (keine Prüfung)

    for(var i = 0; i <= FarbSchritte; ++i)
    {

```



```

        var RotAnteil = Math.floor( (RotAnteil_Start * ((FarbSchritte - i) / FarbSchritte))
                                     + (RotAnteil_End * (i / FarbSchritte))
                                   );

        var GruenAnteil = Math.floor( (GruenAnteil_Start * ((FarbSchritte - i) / FarbSchritte))
                                       + (GruenAnteil_End * (i / FarbSchritte))
                                       );

        var BlauAnteil = Math.floor( (BlauAnteil_Start * ((FarbSchritte - i) / FarbSchritte))
                                      + (BlauAnteil_End * (i / FarbSchritte))
                                      );

        document.bgColor =   "#"
                             + DezimalZuHexaString(RotAnteil)
                             + DezimalZuHexaString(GruenAnteil)
                             + DezimalZuHexaString(BlauAnteil);
    }

    //      Nachfolgende Funktionen existieren NUR, um die globalen Variablen für BODY verfügbar zu machen
    //      da in der HTML-Body-Anweisung keine Javascript-Variablen akzeptiert werden
    function EinAusBlenden_DokumentLaden()
    {
        BackgroundEinAusBlenden(RotAnteil_Start,GruenAnteil_Start,BlauAnteil_Start,
                                RotAnteil_End,GruenAnteil_End,BlauAnteil_End,
                                FarbSchritte
                                );
    }

    function EinAusBlenden_DokumentEntladen()
    {
        //      ist das Laden rückwärts
        BackgroundEinAusBlenden(RotAnteil_End,GruenAnteil_End,BlauAnteil_End,
                                RotAnteil_Start,GruenAnteil_Start,BlauAnteil_Start,
                                FarbSchritte
                                );
    }
}
// -->
</SCRIPT>
</HEAD>
<BODY BGCOLOR=#ffffff   onload="EinAusBlenden_DokumentLaden();"
                        onunload="EinAusBlenden_DokumentEntladen();"
>
</BODY>
</HTML>

```

Beispiel für Dokumenten-Hintergrund auswählen:

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
//      Neues Fenster öffnen mit selektierten Hintergrund
//
function HintergrundSetzen(bgname)
{
    var Fenster=window.open("", "", "width=380,height=380");
    Fenster.focus();
    Fenster.document.open();
    Fenster.document.write( "<HTML><HEAD></HEAD>"
                            + "<BODY BACKGROUND="
                              + ""
                              + bgname
                              + ""
                              + ">"
                              + "</BODY>"
                            + "</HTML>");
    Fenster.document.close();
}
//-->
</SCRIPT>

```



```

</HEAD>
<BODY>
  <A HREF="getbackg.htm" onclick="HintergrundSetzen('bg1.jpg');return false">
    <IMG SRC="bg1.jpg" BORDER="0" WIDTH="96" HEIGHT="96">
  </A>
  <A HREF="getbackg.htm" onclick="HintergrundSetzen('bg2.jpg');return false">
    <IMG SRC="bg2.jpg" BORDER="0" WIDTH="96" HEIGHT="96">
  </A>
</BODY>
</HTML>

```

Beispiel für Änderung der Hintergrundfarben:

```

<HTML>
<BODY bgColor="0000ff"
  onBlur="document.bgColor='ff0000'"
  onFocus="document.bgColor='0000ff'"
>
Dieses Fenster &uuml;ndert die Hintergrundfarbe, wenn es nicht mehr aktiv ist.
</BODY>
</HTML>

```

Zugriff:

window.document.eigenschaft	oder	document.eigenschaft	oder	eigenschaft
window.document.methode()	oder	document.methode()	oder	methode()

```

logischer_window_name.document.eigenschaft
logischer_window_name.document.methode()

```

```

logischer_window_name      laut open()

```

Der **logische** Fenstername muss nur dann kodiert werden, wenn der Bezug nicht auf das Dokument im aktuellen Fenster gehen soll.

Der logische Fenstername stammt aus

der Methode .open() zum window Objekt und kann ein frei festgelegter oder vordefinierter Name sein aus dem Wert des ID-Attributes bei FRAMESET mit FRAMES.

Die Kodierung des Zeigers von document kann dann entfallen, wenn das aktuelle Dokument referenziert wird.

Allerdings ist von der Kodierung ohne window bzw. ohne window.document abzuraten, denn es könnten gleichnamige Eigenschaften und Methoden geben, die in anderen Objekten auch implementiert sind es wird damit die Browserperformance gesenkt.

Beispiel:

```

function OnClickHandler()
{
  alert(FensterZeiger.ID_TextArea.value;
  FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette=  '<HTML>'
           + '<HEAD></HEAD>'
           + '<BODY >'
           + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
           + '<BR>'
           + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
           + ' onclick="opener.OnClickHandler();"'
           + '>'
           + '</BODY>'
           + '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();

```

Eigenschaften:

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:



Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert, wenn gleiche Attributnamen/Eigenschaften betroffen sind.

`.activeElement` Referenz auf dasjenige Objekt im Dokument, das Focus hat
 IE unter 5.x Zeiger auf BODY-Objekt
 ab IE 5.x null

Beispiel:

```
var Zeiger = document.activeElement
```

`.alinkColor` Farbe aller aktiven Links im Dokument
`document.body.alinkColor`
 "#rrggbb" oder vordefinierte Farbname

`.bgColor` deprecated und durch STYLE-Attribut zu ersetzen
 Hintergrundfarbe des Objektes bzw. Dokumentes

`.charset` Zeichensatz zum Encoden eines Objektes im Dokument

`.compatMode` Kompatibilität des IE 6.x zu CSS1 bzw. seinen Browservorgängern
 siehe style Objekt und Kodierung von !DOCTYPE
 ab IE 6.x

Achtung: Das Objekt `document.body` ist nicht mehr verfügbar und wurde durch das Objekt `document.documentElement` ersetzt !

`document.documentElement` ist der Zeiger auf den Body und nicht mehr `document.body` !

Beispiel zur Abfrage auf CSS1-Konformität und damit zur Verwendung von `document.body` bzw. `document.documentElement`

```
function DocTypeAbfrage()
{
    // Annahme: CSS1-Standard, also !DOCTYPE wurde kodiert
    var ZeigerAufDokument=document.documentElement;

    if (document.compatMode=="BackCompat")
    {
        // nicht CSS1-Standard, also wurde kein !DOCTYPE kodiert
        ZeigerAufDokument =document.body;
    }

    return ZeigerAufDokument;
}
```

Hinweise: `document.compatMode` ist nur lesbar, also !DOCTYPE nicht setzbar

Wert "CSS1Compat" ab IE 6
 geliefert wenn !DOCTYPE kodiert wurde
 Rendern mit CSS1 Standard (Standard-Compilant-Modus)
 Wert "BackCompat" ab IE 3
 Geliefert wenn kein !DOCTYPE kodiert wurde
 Rendern mit Nicht-kein Standad-Compilant-Modus

`.cookie` Cookie des Dokumentes als Stringwert

Beispiel:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

    function LeseCookieWert(Name)
```




```
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while ( ( !Gefunden )
        && ( Index < AnzahlCookies)
    );

    return CookieWert;
}
</SCRIPT>
```

.defaultCharset regionaler Standardzeichensatz (Charakter-Set) zum Encoden eines Objektes im Dokument

.designMode Editierbarkeit des Dokumentes (Manipulierbarkeit)
z.B. per Ausführung von Javascript

Syntax:

```
document.designMode [ = Kette ]
[ var Kette = ] document.designMode
```

Kette String

"On"

Dokument kann editiert werden
es wird kein Script ausgeführt !!

"Off" oder "Inherit"

Standard
Dokument kann nicht editiert werden
Scripte werden ausgeführt
"Inherit" bedeutet leider **nicht**: von Eltern
geerbt, da Dokument keine
Eltern haben muss
(z.B. hat Frameset-Dokument
Eltern)

lesen und schreiben

.dir

Umflussrichtung

.doctype

Referenz auf Dokumenttyp

.documentElement

Referenz auf Wurzelknoten (Root Node) des Dokumentes liefern

Beispiel:

```
<SCRIPT>
function HoleHTML()
{ID_Textarea.value= document.documentElement.innerHTML;}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS = 50 ROWS = 10>
```



</TEXTAREA>

Achtung: Das Objekt document.body ist nicht mehr verfügbar und wurde durch das Objekt document.documentElement ersetzt !

document.documentElement ist der Zeiger auf den Body und nicht mehr document.body !

Beispiel zur Abfrage auf CSS1-Konformität und damit zur Verwendung von document.body bzw. document.documentElement

```
function DocTypeAbfrage()
{
    // Annahme: CSS1-Standard, also !DOCTYPE wurde kodiert
    var ZeigerAufDolument=document.documentElement;

    if (document.compatMode=="BackCompat")
    {
        // nicht CSS1-Standard, also wurde kein !DOCTYPE kodiert
        ZeigerAufDolument =document.body;
    }

    return ZeigerAufDolument;
}
```

Hinweise: document.compatMode ist nur lesbar, also !DOCTYPE nicht setzbar

Wert "CSS1Compat" ab IE 6
geliefert wenn !DOCTYPE kodiert wurde
Rendern mit CSS1 Standard (Standard-Compilant-Modus)

Wert "BackCompat" ab IE 3
Geliefert wenn kein !DOCTYPE kodiert wurde
Rendern mit Nicht-kein Standad-Compilant-Modus

.domain Domain-Suffix des Dokumentes
Kommunikation von Dokumenten verschiedener Urls mit gemeinsamen Domainsuffix

Beispiel:

home.test.com und **www.test.com** können kommunizieren,
wenn Kette auf "**test.com**" gesetzt wurde in den Dokumenten beider Urls

.expando Wirksamkeit von Attributen im Dokument ein/aus
true ein, Default
false aus

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <SPAN ID="ID_Span" UNSELECTABLE="on">
        Dieser Text ist <B>selektierbar !!<B>
    </SPAN>
</DIV>
</BODY>
</HTML>
```

.fgColor Vordergrundfarbe (Textfarbe) im Dokument
#rrggbb Standard ist #000000
vordefinierter Farbname (browserspezifisch)

.fileCreatedDate Datum der Dokumenterstellung

Beispiel 1:
"Monday, December 08, 2000"

Beispiel 2:

```
<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);
```



```

var DatumHeute = new Date();

var TagesDifferenz = ( DatumHeute.getTime()
                      - DatumDokumentErzeugung.getTime()
                      )
                    / AnzahlMillisekundenProTag;

alert( "Dokument erzeugt am " + DatumDokumentErzeugung
      + "\n..... also vor " + parseInt(TagesDifferenz) + " Tagen"
      );
}
</SCRIPT>

```

.fileModifiedDate	Datum der letzten Dokumentveränderung
.fileSize	Größe des Dokumentes
.implementation	Referenz auf Objekt implementation zum Dokument
.lastModified	Datum der letzten Dokumentveränderung
.linkColor	Farbe von noch nicht benutzten Links im Dokument "rrggbb" Standard ist "#0000FF" vordefinierter Farbname (browserspezifisch)
.location	Zeiger auf das gleichnamige Objekt

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT>
function LocationAendern()
{
    for(i=0;i<document.all.length;i++)
    {
        if (document.all[i].tagName=="IFRAME")
        {document.all[i].contentWindow.location = "http://www.test.com";}
    }
}
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="LocationAendern();">Location aendern</BUTTON>
<IFRAME SRC="http://www.test.de"></IFRAME>
</BODY>
</HTML>

```

Beispiel 2:

```

<HEAD>
<SCRIPT>
function Entfernen()
{
    // versuche den Text zu entfernen
    try
    {
        var KindZeigerAufTextImDiv = ID_Div.children(0);
        ID_Div.removeChild(KindZeigerAufTextImDiv);
        // Achtung: Der Text ist noch sichtbar !!!!
    }
    // oder fange das Ereignis des bereits entfernten Textes ein
    // und behandle das Ereignis
    catch(x)
    {
        alert( "Text wurde entfernt !\n"
              + "Das Dokument muss neu geladen werden !"
              );
        document.location.reload();
    }
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div" onclick="Entfernen()">
    Klick, um diesen Text zu entfernen !
</DIV>

```



</BODY>	
.parentWindow	Referenz auf Elternfenster des Dokumentes
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern nur lesen bei Objekten document img location
Beispiel: location.protocol liefert z.B. http: und immer inklusive dem Doppelpunkt	
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
Beispiel: alert(document.body.readyState);	
.referrer	Url der direkt vorhergehenden Seite der aktuellen Seite aktuelle Seite muss durch Link angesprungen sein Eingabe in Adresszeile oder per Menü Datei-Öffnen etc. nicht verwendet ist Leerkette, wenn aktuelle Seite nicht durch Link von der vorherigen aktiviert wurde
.title	Fenstertitel des Dokumentes siehe Objekt document

Beispiel für permanenten Fenstertitel-Wechsel:

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    var titel_texte_feld = [
        "Titel 1",
        "Titel 2",
        "Titel 3",
        "Titel 4"
    ];

    var wechsel_tempo = 2500;    // Zeit in ms zwischen zwei Schritten
    var zahler = 0;
    var id;

    function start()
    {
        titel_texte_feld[titel_texte_feld.length] = document.title;
        // beim ersten Aufruf wird der Original-Titel mit gespeichert
        // Länge ab 1, aber Index ab 0
        // Länge == Index +1, an desssen Position der Original-Titel abgelegt wird

        window.setTimeout("wechseln()", wechsel_tempo);
    }

    function wechseln()
    {
        document.title = titel_texte_feld [zahler];
        zahler ++;

        if(zahler >= texte.length)
        { zahler = 0 }

        id = window.setTimeout("wechseln()", wechsel_tempo);
    }
}
-->
</SCRIPT>
```

Beispiel für Fenstertitelzeile mit scrollendem Text:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
//
```



```
//      In der aktuellen Fenster-Titelzeile einen freien Text durch endloses Scrollen von links nach rechts vorsetzen

//*****
//
//      nachfolgende Variablen sind vom Programmierer zu setzen
//
//*****
var VorsetzText_Wert      = "Freien Text vorsetzen durch scrollen und warten ";
var ScrollGeschwindigkeit = "150"; // in Millisekunden, immer > 0, wird nicht geprüft
var DummyLeerzeichenAnzahl = 10; // immer > 0, wird nicht geprüft
// Die Leerzeichen werden nicht sichtbar angezeigt, aber gescrollt.
//      So entsteht eine Wartezeit (pro Leerzeichen laut
//      ScrollGeschwindigkeit)
//      für die vollständige Anzeige von VorsetzText NACH dem
//      vollständigem Vorsetzen per Scrollen von links nach
//      rechts, ehe der Scroller wieder von vorn beginnt

//*****
//
//      nachfolgender Code darf nicht verändert werden
//      wird anstelle von onLoad innerhalb BODY verwendet
//
//*****
//      rechtsbündige Auffüllung mit DummyLeerzeichen
for (i=0; i <=DummyLeerzeichenAnzahl; i++)
{VorsetzText_Wert+=" ";}

//      Länge NACH Auffüllung ermitteln
var VorsetzText_Laenge=VorsetzText_Wert.length;

//      globale Variablen für EndlosScrollen(), die dort laufend manipuliert werden,
//      also hier initialisiert werden müssen
var Zahler="0";
var VorsetzText="";
//-->
</SCRIPT>
</HEAD>

<BODY>
<SCRIPT LANGUAGE="JavaScript1.2">

//*****
//
//      nachfolgender Code darf nicht verändert werden
//
//*****

function EndlosScrollen()
{
    // endloses Scrollen

    if (      (document.all)
        || (document.getElementById)
    )
    {
        // VorsetzText um nächsten Buchstaben aus dem VorsetzText_Wert erweitern
        // (einschliesslich DummyLeerzeichen)
        VorsetzText+=VorsetzText_Wert.charAt(Zahler); // charAt ab Null

        // Fenster-Titelzeile neu belegen
        document.title=VorsetzText;

        Zahler++;
        if (Zahler==VorsetzText_Laenge)
        {
            Zahler="0";
            VorsetzText="";
        }

        setTimeout("EndlosScrollen()",ScrollGeschwindigkeit);
    }
}
```



```

    }

```

window.onload=EndlosScrollen; // ohne () kodieren, damit nicht sofort sondern beim Laden ausgeführt wird

```

//-->
</SCRIPT>
</BODY>
</HTML>

```

.uniqueID durch den Browser automatisch-genriertes ID des Objektes
 Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
 kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

Beispiel:

```

<PUBLIC:ATTACH EVENT="onload" FOR="window" ONEVENT="init()"/>
<SCRIPT LANGUAGE="JScript">
    function init()
    {
        var newTextAreaID = element.document.uniqueID;
        element.document.body.insertAdjacentHTML ( "beforeEnd",
            "<P><TEXTAREA STYLE='height: 200 ;'"
            + "width: 350' ID=" + newTextAreaID
            + "></TEXTAREA></P>"
        );
    }
</SCRIPT>

```

.URL Url des Dokumentes
 Achtung: Gross-und kleinschreibung wird unterschieden !!!
 ist identisch mit location.href

.URLUnencoded Url des Dokumentes ohne Zeichenencoding

.vlinkColor Farbe bereits geklickter Links

.XMLDocument Referenz auf XML-Dokument (XML-DOM)

.XSLDocument Referenz auf den obersten Knoten des XSL-Dokumentes (Style-Sheet-Dokument)

Methoden:

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
 NICHT verkettet sondern in **Zufallsfolge**, es sei denn
 die Handler prüfen ihre Aufruffolge (muss programmiert werden)

Beispiel:

```

<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>
<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor      = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu);           // nicht () kodieren !!!
        detachEvent ('onmouseout', CursorNormal);         // nicht () kodieren !!
    }

    attachEvent ('onmouseover', CursorNeu);
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```



.clear() Dokument löschen

.close() Ausgabe-Datenstream schliessen und Anzeige des Dokumentes beenden
schliessen einer mit .open() erzeugten Dokument-Instanz

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function ErzeugeZurLaufZeit()
    {
        // Dokumentinstanz erzeugen
        var ID_Dokument = document.open("text/html", "replace");

        // Dokumentinhalt festlegen und anzeigen
        ID_Dokument.write("<HTML><BODY>Hallo</BODY></HTML>");

        // Dokumentinstanz schliessen
        ID_Dokument.close();
    }
</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE="button" onclick=" ErzeugeZurLaufZeit();">
</BODY>
</HTML>
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ ' onclick="opener.OnClickHandler();"'
+ '>'
+ '</BODY>'
+ '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.createAttribute() ein Attribut im Dokument erzeugen und Referenz auf das erzeugte Attribut liefern
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
DOM nicht geändert

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function Hinzufuegen()
    {
        // Attribut mit Wert erzeugen
        var ZeigerAufAttribut = document.createAttribute("title");
        ZeigerAufAttribut.value = "Tooltip-Text ";

        // Attribut als Feldelement anhängen
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
    }
}
```



```

</SCRIPT>
</HEAD>
<BODY onload=" Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

`.createComment()` Comment-Objekt (Kommentar-Objekt) im Dokument erzeugen und Referenz liefern
verwendbar anstelle von direkt im HTML- bzw. Script-Quellcode kodiertem Kommentar
DOM wird geändert, da das Dokument erweitert wird

Beispiel:

```
var Kommentar = document.createComment("Das ist ein Kommentar");
```

`.createDocumentFragment()` erzeugt neues Dokument

`.createElement()` HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern
Achtung: Erzeugtes Objekt muss in DOM noch per Methode `.insertBefore()` bzw. `.appendChild()` eingereiht werden.
Hinweis: Attribute mit der Methode `.createAttribute()` erzeugen
DOM wird geändert

Beispiel 1:

```

<SCRIPT>
function Erzeugen()
{
    ID_Span.innerHTML="";
    var FeldDerZeigerAufOption = ID_Select.options[ID_Select.selectedIndex];

    if(FeldDerZeigerAufOption.text.length>0)
    {
        var ZeigerAufElement=
            document.createElement(FeldDerZeigerAufOption.text);
        eval(
            " ZeigerAufElement."
            + FeldDerZeigerAufOption.value
            + "="
            + ID_Input.value
            + ""
        );

        if(FeldDerZeigerAufOption.text=="A")
        { ZeigerAufElement.href="javascript:alert('A link.');" }

        ID_Span.appendChild(ZeigerAufElement);
    }
}
</SCRIPT>
<SELECT ID="ID_Select" onchange="Erzeugen()">
    <OPTION VALUE="innerText">A
    <OPTION VALUE="value">&lt;INPUT TYPE="button"&gt;
</SELECT>
<INPUT TYPE="text" ID="ID_Input" VALUE="Beispiel Text">
<SPAN ID="ID_Span" ></SPAN>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
function Erzeuge()
{
    var Zeiger = document.createElement(
        "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Eins'>"
    );
    document.body.insertBefore(Zeiger);

    Zeiger = document.createElement(
        "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Zwei'>"
    );
    document.body.insertBefore(Zeiger);
}
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE="BUTTON"

```




```

        ONCLICK=" Erzeuge()"
        VALUE="Zwei Radio Buttons erzeugen">
<BR>
<INPUT TYPE="BUTTON"
        ONCLICK="alert(document.body.outerHTML)"
        VALUE="Click um HTML zu sehen">
<BODY>
</HTML>

```

`.createEventObject()` Event-Objekt im Dokument erzeugen nur für Methode `.fireEvent()`

Beispiel:

```

var EventObjekt = document.createEventObject();
document.fireEvent("onclick", EventObjekt);

```

`.createStyleSheet()` styleSheet Objekt im Dokument erzeugen

Beispiel:

```

document.createStyleSheet('styles.css');

```

`.createTextNode()` Text-Objekt im Dokument erzeugen
nur Plain-Text, also keine HTML-Tags

Beispiel:

```

<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>

```

`.detachEvent()` Abschalten des Registrieren eines Events durch Eventhandler
wobei Registrierung mit Methode `.attachEvent()` aktiviert wurde
Abschalten = ordnet dem Window-Objekt das Event laut Parameter `event_bezeichner`
zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt
(also Standardbehandlung aktiv)

Beispiel:

```

<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
function CursorNeu()
{
    if (event.srcElement == element)
    {
        normalColor      = style.color;
        runtimeStyle.color = "red";
        runtimeStyle.cursor = "hand";
    }
}

function CursorNormal()
{
    if (event.srcElement == element)
    {
        runtimeStyle.color = normalColor;
        runtimeStyle.cursor = "";
    }
}

function EventEntfernen()
{
    detachEvent ('onmouseover', CursorNeu);           // nicht () kodieren !!!
    detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
}

attachEvent ('onmouseover', CursorNeu);
attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```



`.elementFromPoint()` liefert Referenz auf Objekt an Pixelpos relativ zum Fenster
Fenster linke obere Ecke (0,0)
Objekt muss Mausevents unterstützen

`.execCommand()` Kommando ausführen z.B. im aktuellen Dokument
in aktueller Selektion
im aktuellen Bereich
erst nach dem kompletten Laden des Dokumentes zulässig
Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
Control = Element zur Steuerung analog zum HTML-Element (Tag)
Input-Control = Element mit Eingabeeigenschaft

Beispiel 1:

```
<HTML>
<BODY>
  <H1 UNSELECTABLE="on">Demo</H1>
  <SCRIPT>
    function AddLink()
    {
      var SelektierterText = document.selection.createRange();

      if (!SelektierterText == "")
      {
        // Link erzeugen
        document.execCommand("CreateLink");

        if (SelektierterText.parentElement().tagName == "A")
        {
          // markierten Text mit Eltern-Url ersetzen
          SelektierterText.parentElement().innerText=
            SelektierterText.parentElement().href;

          // Vordergrundfarbe setzen im Dokument
          document.execCommand(
            "ForeColor", "false", "#FF0033");
        }
      }
      else
      { alert("Bitte im blauen Text selektieren !"); }
    }
  </SCRIPT>
  <P UNSELECTABLE="on">
    Selektiere (markiere) im nachfolgenden blauen Text die Stelle mit
    dem Text MARKIERE_MICH.<BR>
    Danach auf das Button klicken.<BR>
    Anstelle von MARKIERE_MICH im blauen Text erscheint dort
    nun eine Url.
  </P>
  <P STYLE="color=#3366CC">
    Meine beliebteste Webseite MARKIERE_MICH bitte besuchen !
  </P>
  <BUTTON onclick="AddLink()" UNSELECTABLE="on">Klick</BUTTON>
</BODY>
</HTML>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
  function HandlerFuerOnMoveStart()
  {
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
  }

  function HandlerFuerOnMove ()
  {
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
```



```

        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandlerFuerOnMoveEnd()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    // 2-D Positionierung einschalten
    document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offserTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

.focus()

Focus setzen und Focus-Event auslösen
 nur nach dem kompletten Laden des Dokumentes
 vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

Beispiel:

```
<BODY onload="document.body.focus();>
```

.getElementById()

Referenz auf das im Dokument ZUERST gefundene Objekt laut ID (analog zum ID-Attribut) liefern
 Achtung: Objekte, die kein ID besitzen, werden nicht erfasst !

Für Verwaltung per NAME (analog zum NAME-Attribut):
 siehe Methode getElementByName()

Für Verwaltung per Tag-Name
 siehe Methode .getElementsByTagName()

wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenziert

Eine Referenzierung einer Collection der Objekte mit gemeinsamen ID ist leider nicht möglich. Deswegen der strenge Hinweis:

In der Regel werden ID vom Programmierer objektweise getrennt vergeben, es sei denn, man will bewusst eine Gruppe von Objekten (z.B. RadioBox) verwaltbar machen und kennt diese Objekte. Die maschinelle Analyse eines fremden Dokumentes mit der Methode .getElementById() ist nicht möglich.

DOM nicht geändert

Beispiel:

```

<SCRIPT>
function Referenziere()
{
    var Zeiger=document.getElementById("ID_Div");
}
</SCRIPT>
<DIV ID="ID_Div">Test</DIV>
<INPUT TYPE="button" VALUE=" Referenziere" onclick=" Referenziere()">

```

.getElementsByTagName()

Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME (analog zum Attribut NAME) liefern

Achtung: Objekte, die kein NAME besitzen, werden nicht erfasst !

Für Verwaltung per ID (analog zum ID-Attribut):
 siehe Methode getElementById()

Für Verwaltung per Tag-Name
 siehe Methode .getElementsByTagName()

DOM nicht geändert



Beispiel:

```
<SCRIPT>
    function Referenziere()
    {
        var FeldDerInput =document.getElementsByName("GemeinsamerName");
    }
</SCRIPT>
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="button" NAME="Button" VALUE=" Referenziere" onclick=" Referenziere()">
```

`.getElementsByName()` Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.
 Hinweis: Natürlich kann auch das `document`-Objekt so verarbeitet werden (beachte dabei `document.all` Collection)
 Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !
 Für Verwaltung per ID (analog zum ID-Attribut):
 siehe Methode `getElementById()`
 Für Verwaltung per NAME (analog zum NAME-Attribut):
 siehe Methode `getElementsByName()`
 DOM nicht geändert

Beispiel 1:

```
var DIV_KinderZeigerFeld = document.body.getElementsByTagName("DIV");
```

Hinweis: entspricht `var DIV_KinderZeigerFeld = document.body.all.tags("DIV");`

Beispiel 2:

```
<SCRIPT>
    var Feld_Span = ID_DivEltern.getElementsByTagName("SPAN");
    // alle SPAN-Kinder referenzieren
</SCRIPT>
<DIV ID="ID_DivEltern">
    <SPAN>
        Span-Kind von ID_DivEltern
    </DIV>
        Div-Kind vonDivEltern-Span
        <SPAN>
            Span-Kind vonDivEltern-Span-Div
            <SPAN>
        </DIV>
    </SPAN>
</DIV>
```

Beispiel 3:

```
<SCRIPT>
    function Anzeige()
    {
        var ZeigerAufOnClickEventQuelle=event.srcElement;
        var Feld =
            ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
        alert(
            "Anzahl LI : "
            + Feld.length
            + "\nErster Eintrag: "
            + Feld [0].childNodes[0].nodeValue
        );
    }
</SCRIPT>
<UL onclick="Anzeige()">
    <LI>Menuepunkt 1
    <UL>
        <LI> Menuepunkt 1.1
        <OL>
            <LI> Menuepunkt 1 1.1
            <LI> Menuepunkt 1 1.2
        </OL>
        <LI> Menuepunkt 1.2
        <LI> Menuepunkt 1.3
    </UL>
    <LI> Menuepunkt 2
    <UL>
        <LI> Menuepunkt 2.1
```



```
</LI> Menüepunkt 2.3
</UL>
<LI> Menüepunkt 3
</UL>
```

<code>.hasFocus()</code>	Objekt im Focus-Zustand
<code>true</code>	Objekt hat den Focus
<code>false</code>	Objekt hat keinen Focus

<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
Attribute sind:	HTML
	Events
	Styles
	ab IE 5.01 auch ID, NAME
	Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
	DOM wird geändert

Beispiel:

```
<SCRIPT>
    function Mischen()
    { ID_SPAN.children[1].mergeAttributes(ID_SPAN.children[0]); }
</SCRIPT>
<SPAN ID=ID_SPAN>
    <DIV      ID="ID_Div_Quelle"
              ATTRIBUTE1="true"
              ATTRIBUTE2="true"
              onclick="alert('click');"
              onmouseover="this.style.color='#0000FF';"
              onmouseout="this.style.color='#000000';"
    >
        Quell<B>Div</B>
    </DIV>
    <DIV      ID="ID_Div_Ziel">
        Ziel-Div
    </DIV>
</SPAN>

<INPUT TYPE="button" VALUE=" Mischen" onclick="Mischen()">
```

<code>.open()</code>	Dokument instanzieren und mit/ohne Fenster öffnen und Referenz liefern auf Dokument
	Ausgabe-Datenstream öffnen und Anzeige des Dokumentes
	Instanzieren ohne <code>.open()</code> : siehe <code>.write()</code> und <code>.writeln()</code>
	Hinweis: neues Fenster erzeugen als unterstes der aktuellen Fensterhierarchie
	und dann Fenster öffnen (anzeigen, rendern)

Syntax:

für Öffnen mit Fenster:

[illegible]

für Öffnen im aktuellen Fenster

```
[ var ZeigerAufDokument =] document..open([mime_typ] [,Replace])
```

für Öffnen im beliebigen Fenster

```
[ var ZeigerAufDokument =] window_instanz.document.open([mime_typ] [,Replace])
```

URL	String mit Multipurpose Internet Mail Extensions (MIME) Typ Standard: MIME-Typ "text/html" nur "text/html" zulässig
-----	---

Name	String Dokumentname für Verlauf (History)
	wenn kodiert, so immer das aktuelle Dokument durch das Neue ersetzt im Verlauf
	wenn nicht kodiert, so neues Dokument angehängt an History

physischer Window-Name: identisch mit Wert laut Attribut
TARGET z.B. im Link

dient der Referenz
auch null kodierbar für keine Referenz

Vordefinierte Namen:

```
"_blank"
" media"
```

Standard : ohne Name

Dokument laut Url wird in den HTML-Content-Bereich der Media Bar geladen



ab IE 6.x

"_parent" Dokument laut Url wird in den Elternframe geladen wird.

wenn kein Frame so wirkungslos

"_search" Dokument laut Url wird in das Browser-Search-Fenster geladen ab IE 5.x

"_self" Dokument laut Url wird in das neue Fenster geladen

"_top" Dokument laut Url wird in das oberste Fenster geladen
identisch mit _self, da neues Fenster das oberste wird

mime_type Datentyp des Dokumentes
meistens **"text/html"** HTML-Dokument
oder **"text/plain"** Normaltext-Dokument

Features String als Parameterliste mit Kommatrennung
(Liste der Fensterkomponenten)
gesamte Liste in " " bzw. ' ' setzen
z.B. "fullscreen=yes, toolbar=yes"
gesamte Liste ist 1 Zeichenkette,
die in 1 Quelltextzeile passen muss,
oder in Teilketten zerlegt mit dem
+ Operator zusammengesetzt wird
Blanks in Liste **nicht** zulässig

Listenelement: option=wert
wenn mindestens 1 Element kodiert, so alle
anderen nicht kodierten Elemente
automatisch deaktiviert, also immer alle
gewünschten Optionen kodieren !!!
Standardwert eines Merkmales, wenn
Liste kodiert wurde: no oder 0
Liste nicht kodiert wurde: yes oder 1
keine Standardwerte vorhanden für Pixelangaben
da diese vom Browser automatisch
belegbar sind

alwaysLowered = { yes | no }
yes: Fenster öffnen und permanent als
unterstes in der Fensterhierarchie
belassen
nur per signiertem Script
beachte .setZOptions()
nur NS

alwaysRaised = { yes | no }
yes: Fenster öffnen und permanent als
oberstes in der Fensterhierarchie belassen
nur per signiertem Script
beachte .setZOptions()
nur NS

channelmode = { yes | no | 1 | 0 }
default ist no bzw. 0
yes oder 1 für Channelleiste anzeigen
(Fenster im Channel-Mode anzeigen)
nur IE

dependent = { yes | no }
yes: Fenster an den .opener bezüglich
Fensterschliessen koppeln: auch
schliessen, wenn .opener geschlossen
wird
sowie geöffnetes Fenster nicht in
der Taskleiste von Windows
anzeigen
nur NS

directories = { yes | no | 1 | 0 }
default ist yes bzw. 1



yes oder 1 Directory Buttons anzeigen

fullscreen = { yes | no | 1 | 0 }
 default ist no bzw. 0
 yes bzw. 1 Vollbild anzeigen also ohne
 Leisten etc., wobei damit fast alle
 Steuerungselemente unsichtbar werden
 nur IE
 Hinweis: ALT+F4 schliesst das Fenster

height = Pixelwert, Fensterhöhe
 bei IE : mindestens 100 (wenn < 100, so auf
 100 automatisch gesetzt)
 bei NS: nur mit signiertem Script < 100
 ohne signiertem Script: wenn < 100,
 so auf 100 automatisch gesetzt

hotkeys = { yes | no }
 yes: alle sicherheitsrelevanten
 Tastenkombinationen
 verwendbar machen
 no: nur noch ALT+F4 funktioniert
 (schliessen des Fensters)
 nur NS

innerHeight= anzeigebereich_höhe_in_pixel
 ohne signiertes Script: >= 100
 Anzeigebereich = Fenster abzüglich
 Leisten, Scrollbars etc.
 nur NS

innerWidth= anzeigebereich_breite_in_pixel
 ohne signiertes Script: >= 100
 Anzeigebereich = Fenster abzüglich
 Leisten, Scrollbars etc.
 nur NS

left = X-Koordinate in Pixels bezüglich linker
 oberer Screen-Ecke (0,0)
 nur IE

location = { yes | no | 1 | 0 }
 default ist yes bzw. 1
 yes bzw 1 für URL-Eingabezeile anzeigen
 (Adresszeile anzeigen)

menubar = { yes | no | 1 | 0 }
 default ist yes bzw. 1
 yes bzw. 1 für Menübar anzeigen
 (Leiste der Pull-Down-Menüs anzeigen)

outerHeight= fenster_höhe_in_pixel
 ohne signiertes Script: >= 100
 Fenster = Anzeigebereich sowie Leisten,
 Scrollbars etc.
 nur NS

outerWidth= fenster_breite_in_pixel
 ohne signiertes Script: >= 100
 Fenster = Anzeigebereich sowie Leisten,
 Scrollbars etc.
 nur NS

personalbar = { yes | no }
 yes: persönliche Toolbar anzeigen
 nur NS

resizable = { yes | no | 1 | 0 }
 default ist yes bzw. 1
 yes bzw. 1 für Größenveränderung des
 Fensters erlaubt per Mausziehen



bei NS beachte .setResizable()

screenX= horizontale_pixel_pos des Fensters
bezüglich dem Bildschirm,
dessen Ursprung (0,0) in der
linken oberen Ecke liegt

nur NS

screenY= vertikale_pixel_pos des Fensters
bezüglich dem Bildschirm,
dessen Ursprung (0,0) in der
linken oberen Ecke liegt

nur NS

scrollbars = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Scrollbalken anzeigen
sobald Fensterinhalt größer als
Anzeigebereich des Fensters

status = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Statuszeile anzeigen

titlebar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Titelzeile anzeigen
Titel werden immer angezeigt bei Dialog-Box
bei NS: nur mit signiertem Script setzbar

toolbar = { yes | no | 1 | 0 }
default ist yes bzw. 1
yes bzw. 1 für Toolbar (Navigationsleiste)
anzeigen (Button Zurück etc.)

top = Y-Koordinate in Pixels bezüglich linker oberer
Screen-Ecke (0,0)
nur IE

width = Pixelwert, Fensterbreite,
bei IE : mindestens 100 (wenn < 100, so auf
100 automatisch gesetzt)
bei NS: nur mit signiertem Script <= 100
ohne signiertem Script: wenn < 100,
so auf 100 automatisch gesetzt

z-lock = { yes | no }
yes: Fenster kann kein anderes
überlagern
nur per signiertem Script
beachte .setZOptions()
nur NS

Replace true, so History-Eintrag des Dokumentes, in dem das neue
Fenster erzeugt wird, ersetzen durch den Eintrag
des neuen Fensters, also Zurück zum Dokument,
dass das Fenster öffnet, nicht möglich
false, so neuen History-Eintrag zum neuen Fenster
erzeugen, also zurück zum alten Fenster möglich

ZeigerAufWindow Referenz (ID) auf das neue Fenster mit Dokument
z.B. für Methode window.close()

ZeigerAufDokument Referenz (ID) auf das neue Dokument
z.B. für Methode ZeigerAufDokument.close()

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
```

```
function ErzeugeZurLaufZeit()
```




```

    {
        // Dokumentinstanz erzeugen
        var ID_Dokument = document.open("text/html", "replace");

        // Dokumentinhalt festlegen und anzeigen
        ID_Dokument.write("<HTML><BODY>Hallo</BODY></HTML>");

        // Dokumentinstanz schliessen
        ID_Dokument.close();
    }
</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE="button" onclick=" ErzeugeZurLaufZeit();">
</BODY>
</HTML>

```

Beispiel:

```

function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= ' <HTML>'
          + '<HEAD></HEAD>'
          + '<BODY >'
          + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
          + '<BR>'
          + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
          + ' onclick="opener.OnClickHandler();" '
          + '>'
          + '</BODY>'
          + '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();

```

.queryCommandEnabled()	prüfen ob Kommando ausführbar ist
.queryCommandIndeterm()	prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState()	Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
.queryCommandSupported()	prüfen ob Kommando im aktuellen Bereich unterstützt wird
.queryCommandValue()	Wert eines Kommandos liefern
.recalc()	dynamischen Eigenschaften des Dokumentes neu berechnen Hinweis: andere Objekte, die Eigenschaften des Dokumentes nutzen, werden auch neu berechnet, wenn Eigenschaften nicht in einem Berechnungs Ausdruck vorliegen

Beispiel 1:

```

<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var seconds = 0;

    function init()
    {
        ID_Div1.style.setExpression("width","seconds*10");
        ID_Div2.setExpression("innerText","seconds.toString()");
    }

    function timer()

```



```

    {
        seconds++;
        document.recalc();
    }

function starttimer()
{
    if (timerID == null)
    {
        timerID = setInterval("timer()", 1000);
        ID_Button1.disabled = true;
        ID_Button2.disabled = false;
    }
}

function stoptimer()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
        ID_Button1.disabled = false;
        ID_Button2.disabled = true;
    }
}

function resettimer()
{ seconds = 0; }
</SCRIPT>
</HEAD>
<BODY onload="init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div2" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"                                onclick="starttimer()">Start Timer</BUTTON></BR>
    <BUTTON ID="ID_Button2" DISABLED="true"                onclick="stoptimer()">Stop Timer</BUTTON><BR>
    <BUTTON                                onclick="resettimer()">Reset Timer</BUTTON><BR>
</BODY>
</HTML>

```

Beispiel 2 für Sekundenbalken:

```

<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

```



```

function Starten()
{
    if (timerID == null)
    {
        // Start-Button nicht aktivierbar machen
        ID_Button1.disabled = true;

        // Stop-Button aktivierbar machen
        ID_Button2.disabled = false;

        // Uhr neu starten
        timerID = setInterval("Uhr()", 1000);
    }
}

function Stoppen()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
        ID_Button1.disabled = false;
        ID_Button2.disabled = true;
    }
}

function ZurueckSetzen()
{
    Zahler = 0;
}
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold" ></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

Beispiel 3 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln

```



```

// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++++ Routinen der Sekundenzählung
var SekundenZahler          = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen()    // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekunden zählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekunden zählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl          = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
                                                    // Timerzeit für Rekursion
    this.SoundFileBeendet      = true; // kein Sound aktiv
}

// ++++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",

```



```

        SoundObjekt.SoundFileDauerInMillisekundeSekunden
    );
}
else
{
    // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

    // Sound zu Ende
    SoundObjekt.SoundFileBeendet=true;

    // Sekundenzähler stoppen
    SekundenZaehlen_Stop();

    // und Meldung
    var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
    var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
    alert(
        "Wiedergabe beendet\nUngenauigkeit des Timers = "
        + TimerUngenauigkeit1.toString()+ " Sekunden\n"
        + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
    );
}
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler        = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
        "SekundenZahler * PixelBreiteProBalkenErweiterung"
    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText = "Der Sound dauert "
        + SoundDauerInSekunden.toString()
        + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write(
        '<BGSOUND ID= "ID_BGSound" LOOP="0">'
    );

    document.write(
        '<DIV ID="ID_DIV_Balken"'
        + 'STYLE="background-color:lightblue"'
        + '>'
    );
}

```



```

        + '</DIV>'
        + '<BR>'
    );

    document.write(
        '<DIV ID="ID_DIV_SekundenZahler"'
        + 'STYLE="color:hotpink;font-weight:bold"'
        + '>'
        + '</DIV>'
        + '<BR>'
    );

    document.write(
        '<DIV ID="ID_DIV_MessLatte"'
        + 'STYLE="color:white;background-color:gray"'
        + '>'
        + '</DIV>'
    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ----- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
    }
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

.releaseCapture() Maus-Überwachung ausschalten für ein Objekt
 Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,
 onmouseover und onmouseout.
 Hinweis: einschalten per Methode **.setCapture()**

Beispiel:

```

<BODY onload="ID_Div.setCapture();"
onclick="document.releaseCapture();"
>
    <DIV ID="ID_Div"
        onmousemove="ID_Textarea.value = event.clientX + event.clientY;"
        onlosecapture="alert(event.srcElement.id + ' hat keine Mausueberwachung mehr');"
    >
        <TEXTAREA ID="ID_Textarea" COLS=2>Test</TEXTAREA>
    </DIV>
</BODY>

```

.setActive() Objekt für die Eventdurchreichung aktivieren
 aber ohne es zu fokussieren
 und ohne es scrollbar zu machen

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    var ID_Fenster;

    function FensterErzeugen()
    {
        ID_Fenster = window.open( "/test /test.htm",
                                " ID_Fenster",
                                "top=10px,left=480px,height=375px,width=200px,resizable=1"
                                );
        this.focus();
    }

    function ButtonAktivieren()
    {window.parent.ID_Fenster.ID_Button.setActive();}

</SCRIPT>
</HEAD>
<BODY onload="FensterErzeugen();">

```



```

<BUTTON ID="ID_Button" onclick="ButtonAktivieren();">
    Button aktivieren
</BUTTON>
</BODY>
</HTML>

```

.write()

Text bzw. HTML-Ausdruck in das Dokument ausgeben und anzeigen bzw. sofort parsen

Hinweis:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen** HTML-Dokumentes, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

Plain-Text

HTML-Text

Script

.writeln()

Text bzw. HTML-Ausdruck in das Dokument ausgeben und anzeigen bzw. sofort parsen

Hinweis:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen** HTML-Dokumentes, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

nach der Anzeige einen Zeilenvorschub anzeigen

Plain-Text

HTML-Text

Script

4.3.2.2.4.1. *window.document.all Collection des Internet Explorer* (dokumentbezogene Verwaltung von HTML-Elementen)

Die Collection ist

das Feld der Zeiger **aller** HTML-Elemente des Dokumentes inklusive der HTML-Tags HTML, HEAD, TITLE etc.

nicht das Feld aller Knoten im DOM (siehe Beschreibung vom DOM), weil die Folge der HTML-Elemente im HTML-Code zum Dokument **verschieden** ist zur Folge in der gesamten **und** dokument-eigenen DOM-Hierarchie.

dient zur Verwaltung aller

IE-üblichen und IE-spezifischen Eigenschaften und Methoden der Objekte vom Objekt document und natürlich der des Dokumentes selbst

Eigenschaften und Methoden und von Objekten außerhalb der Hierarchie des Objektes document

wird **nicht** vom Netscape erkannt

Unterschied der Kodierung document und document.all:

mit oder ohne .all werden kodiert:

IE-**übliche** Eigenschaften und Methoden (also die des Objektes document), da diese vererbt sind Eigenschaften und Methoden von Objekten außerhalb der Hierarchie des Objektes document

mit .all werden kodiert:IE-**spezielle** Eigenschaften und Methoden (also die der Objekte innerhalb der Hierarchie des Objektes document)

Empfehlung: Es sollte beim IE **immer** document.all kodiert werden, egal um welches HTML-Objekt es sich innerhalb des Dokumentes handelt, also auch bei der Referenzierung über den Wert des ID- bzw. NAME-Attributes. Anders formuliert: Es sollte immer der absolute Pfad ab Objekt document kodiert werden, wobei der Pfad bei Mehrfachverwendung in einem Zeiger liegen sollte (höhere Browser-Performance).

IE-übliche bzw. -spezielle Objekte:

IE-**übliche** Eigenschaften und Methoden gehören dem Objekt document

werden an die Objekte des Objektes document vererbt
können wie folgt kodiert werden:

```

document.eigenschaft
document.methode()
document.all.eigenschaft
document.all.methode()

```

oder gehören Objekten außerhalb der Hierarchie des Objektes document
werden z.T. vom Netscape unterstützt

können wie folgt kodiert werden: document.object.eigenschaft



document.object.methode()
mit object als Zeiger auf
das HTML-Element
z.B. laut ID-Attribut

IE-spezielle Eigenschaften und Methoden gehören den Objekten des Objektes document
liegen innerhalb der Hierarchie des Objektes document
müssen mit .all kodiert werden: document.all.eigenschaft
document.all.methode()

Falls es zulässig ist, dass ein Objekt des Objektes document als Kind eines anderen Objektes instanziiert werden kann,
dann ist die Kodierung natürlich um die Referenz auf das Elternobjekt zu erweitern, es sei denn, man verwendet
den Wert des ID-Attributes des Kindes als eindeutigen Zeiger .

Es ist möglich, dass **sämtliche** Eigenschaften und Methoden, deren Kodierung **kein .all** benötigen
(also IE-übliche Eigenschaften und Methoden bzw. Eigenschaften und Methoden von Objekten außerhalb der
Hierarchie des Objektes document) **auch vom Netscape** unterstützt werden (falls das betreffende HTML-Element
unterstützt wird).

Falls der Netscape auch Objekte des IE unterstützt, so kann es trotzdem sein, dass deren Eigenschaften und Methoden im NS
bzw. IE z.T. **verschieden** implementiert sind (inklusive Syntax).

Empfehlung: Es sollte beim IE immer document.all kodiert werden, egal um welches HTML-Objekt es sich innerhalb des
Dokumentes handelt, also auch bei der Referenzierung über den Wert des ID- bzw. NAME-
Attributes. Anders formuliert: Es sollte immer der absolute Pfad ab Objekt document kodiert
werden, wobei der Pfad bei Mehrfachverwendung in einem Zeiger liegen sollte (höhere Browser-
Performance).

Erzeugung:

durch Browser

Syntax:

```
[ var ZeigerAufFeld = ] document.all
[ var ZeigerAufFeldElement = ] document.all[Index [, SubIndex]]
```

Index Integer ab 0
oder String Name oder ID des Elementes
muss in [] kodiert sein

SubIndex optional
nur kodieren wenn Index ein String ist
Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2. HTML-Elemente übergreifende Verwaltung im HTML-Dokument**4.3.2.2.4.2.1. Verwaltung mehrerer HTML-Elemente gleicher Art im HTML-Dokument (Auswahl)**

Die Verwaltung erfolgt z.T. über Collectionen als Zeigerfelder auf Instanzen gleichartiger HTML-Elemente als Objekte.

4.3.2.2.4.2.1.1. window.document.anchors Collection des Internet Explorer (HTML-Element A (Anker))

Feld der Zeiger aller Anker im Dokument

Feldelementefolge laut HTML-Coding

Erzeugung:

durch den Browser

Beispiel für Anker:

```
<A ID="ID_Anker"
  HREF="url"
  NAME="logischer_anker_name"
  TARGET="logischer_window_name"
>
```




```

        anker_text
    </A>

```

Hinweis zu HREF= :

kann leer sein	per	Leerkette ""
	oder	"javascript:void(0);"

zu Anspringen eines Ankers: den Wert laut NAME ablegen in
 window.location.hash ohne vorgesetzten #
 window.location.href mit vorgesetzten #

Syntax:

```

[ var ZeigerAufFeld = ] document.anchors
[ var ZeigerAufFeldElement = ] document.anchors[Index, [ SubIndex]]

```

Index	Integer ab 0
oder	String Name oder ID des Elementes
	muss in [] kodiert sein
SubIndex	optional
	nur kodieren wenn Index ein String ist
	Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement.eigenschaft
 ZeigerAufFeldElement.methode()

document.all.**ID_Anker**.eigenschaft
 document.all.**ID_Anker**.methode()

Beispiel: Alle Textmarken (Anker) eines Dokumentes für deren Anwahl in einem Extra-Fenster anzeigen

```

<HTML>
<HEAD>
    <SCRIPT>
    <!--
        function textmarken_fenster_anzeigen()
        {
            var textmarken_fenster = open( "",
                "Textmarken-Fenster",
                "toolbar=0,location=0,scrollbars=0,menu=0,"
                + "dependent,resizable,status,width=150,height=300"
            );

            with(textmarken_fenster.document)
            {
                // HTML-Fenster formatieren
                window.name = "logischer_fenster_name"
                open("text/html")
                writeln("<HTML><HEAD>")
                writeln(
                    "<TITLE>Steuerung f&uuml;r &quot;"
                    + document.title
                    + "&quot;</TITLE>"
                )
                writeln(
                    "<BASE HREF="
                    + location.href
                    + "' TARGET='logischer_fenster_name'"
                )
                writeln("</HEAD><BODY>")

                // Textmarken (Anker im Fenster anzeigen)
                for(var i = 0; i < document.anchors.length; i++)
                {
                    writeln(
                        "<P><A HREF='#"
                        + document.anchors[i].name
                        + "'>"
                        + document.anchors[i].name
                        + "</A><P>"
                    );
                }

                writeln("</BODY></HTML>");
                close();
            }
        }
    -->

```



```

        focus();
    }
}
// -->
</SCRIPT>
</HEAD>
<BODY>
    <SCRIPT>
    <!--
        document.write(
            "<P><A HREF='javascript: textmarken_fenster_anzeigen ();'>"
            + "Textmarken-Fenster anzeigen</A></P>"
        );
    // -->
</BODY>
</HTML>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

f4.3.2.2.4.2.1.2. window.document.applets Collection des Internet Explorer

Feld der Zeiger aller Applet-Objekte im Dokument

Elementefolge laut HTML-Coding

Applet muss alle Eigenschaften und Methoden, die im HTML-Dokument benutzt werden sollen, als **public** deklarieren

Erzeugung:

durch den Browser

Java-Applet (*.class) in HTML einbinden:

```

<APPLET ID="ID_Applet"
    CODE="class_datei"
    CODEBASE="quellverzeichnis"
    NAME="Name_Applet"
    HEIGHT=applet_fenster_hoehe_in_pixel
    WIDTH=applet_fenster_breite_in_pixel
    MAYSCRIPT=true oder false
    ALT="alternativer_text"
    ALIGN=ausrichtung
    HSPACE=fenster_abstand_links_und_rechts_von_umgebung
    VSPACE=fenster_abstand_loben_und_unten_von_umgebung
>
    <PARAM Name="parameter_name"
        VALUE=parameter_wert
    >
    ....
</APPLET>

```

Syntax:

```

[ var ZeigerAufFeld = ] document.applets
[ var ZeigerAufFeldElement = ] document.applets[Index [, SubIndex]]

```

Index	Integer ab 0 muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

document.all.ID_Applet.eigenschaft
document.all.ID_Applet.methode()

document.all.Name_Applet.eigenschaft
document.all.Name_Applet.methode()



Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge

Methoden:

`.item()` Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit `<INPUT TYPE=image ...>` da dafür die `children`-Collection verwendet werden muss !!!

`.namedItem()` Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

`.tags()` Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe `tags` Collection des DOM

`.urns()` Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.3. window.document.body.timeAll Collection des Internet Explorer

Feld der Zeiger aller per Behavior time2 verwalteten Elemente (getimte Elemente) siehe auch `style.time2` Behavior und `.timeChildren` Collection

Erzeugung:

durch den Browser

Syntax:

```
[ var ZeigerAufFeld = ] document.all.body.timeAll
[ var ZeigerAufFeldElement = ] document.all.body.timeAll [Index]

[ var ZeigerAufFeld = ] document.all.ID_Body.timeAll
[ var ZeigerAufFeldElement = ] document.all.ID_Body.timeAll [Index]
```

Index Integer ab 0
oder String Name oder ID des Elementes
muss in [] kodiert sein

`ZeigerAufFeldElement.eigenschaft`
`ZeigerAufFeldElement.methode()`

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

`.item()` Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit `<INPUT TYPE=image ...>` da dafür die `children`-Collection verwendet werden muss !!!

4.3.2.2.4.2.1.4. window.document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)

Feld der Zeiger aller als Plugin per EMBED-Tag eingebetteten Objekte im Dokument (inkl. Applets) Elementefolge laut HTML-Coding

Achtung: Ab IE 6.x werden keine Plugins mehr unterstützt, damit ist die `plugins` Collection hinfällig. Inwieweit diese Collectionen noch implementiert ist oder bleibt, ist durch den Programmierer zu prüfen. Ab dem IE 6.x muss jedes Plugin, das in das Dokument eingebunden werden soll, durch ein ActiveX-Control implementiert werden. Plugins, wie sie beim Netscape existieren, laufen unter dem IE ab 6.x nicht mehr.

Es ist zu vermuten, dass im IE ab 6.x diese Collection nur eine Dummy-Funktion hat, also existiert, aber nie angefasst wird.

Diese Collection ist ein Alias für die `plugins` Collection **nur** bezüglich von Plugins (und nicht anderen einbettbaren Objekten), wobei letztere nur der Kompatibilität mit anderen plugin-fähigen Browsern dient. Alle eingebetteten Objekte haben in `document.embeds` ihren Zeiger (inklusive Plugins, Ausnahme: siehe tiefer).

Das Ansprechen von Plugins kann über die Collection `navigator.mimeTypes` per Eigenschaft `.enabledPlugin` erfolgen, die einen Zeiger auf das installierte Plugin enthält (wenn nicht installiert, so null.Zeiger). Diese Collection muss aber beim Internet Explorer nicht komplett gefüllt sein, kann aber (ausprobieren). Wenn der Zeiger nicht null ist, dann sind die plugin-eigenen Methoden und Eigenschaften über Punktnotation ansprechbar.

Es ist empfehlenswert, beim Internet Explorer **nur** mit der `document.embeds` Collection zu arbeiten. Die Collection `navigator.mimeTypes` kann zwar unter NS und IE mit dem gleichen Script-Code angesprochen werden, aber das ist spätestens ab IE 6.0 nicht mehr möglich.

Erzeugung:

durch Browser

Beispiel für EMBED-Tag beim IE unter 6.x und beim NS:

```
<EMBED
SRC="url"
TYPE="mime_typ"           // z.B. "image/gif"
PLUGINSOURCE="plugin_url"
HEIGHT="hoehe_plugin_objekt"
WIDTH="breite_plugin_objekt"
NAME="ID_Embed"
HIDDEN="true oder false"  // true so Objekt unsichtbar
```



```
>
    <PARAM Name="parameter_name" VALUE=parameter_wert>
    .....
</EMBED>
```

Syntax:

```
[ var ZeigerAufFeld = ] document.embeds
[ var ZeigerAufFeldElement = ] document.embeds[Index, [SubIndex]]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

```
ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()
```

beim IE:

```
document.all.ID_Embed.eigenschaft
document.all.ID_Embed.methode()
```

beim NS:

```
document.ID_Embed.eigenschaft
document.ID_Embed.methode()
```

Eigenschaften beim Internet Explorer:

```
.length      Anzahl der Feldelemente also Feldlänge
```

Methoden beim Internet Explorer:

```
.item()       Referenz auf Feldelement anhand des Integer-Indexes oder des
               Attributnamen (analog zu ID oder NAME-Attribut) liefern
               außer bei Formular mit <INPUT TYPE=image ...>
               da dafür die children-Collection verwendet werden muss !!!
.namedItem()  Referenz auf Eintrag (FeldElement) anhand des Namen
               (analog zu ID oder NAME-Attribut) liefern
.tags()       Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
               siehe tags Collection des DOM
.urns()       Referenz auf Feld aller Elemente mit gemeinsamer URN liefern
```

4.3.2.2.4.2.1.5. window.document.form.elements Collection des Internet Explorer

Felder der Zeiger auf alle Formularkomponenten **außer** Objekt input.image, das damit **im** Formular nicht referenzierbar wird:
Für input image ist immer die children Collection zu verwenden.

Reihenfolge der Feldelemente laut der Kodierung der Elemente im Formular

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_form_objekt.elements
[ var ZeigerAufFeldElement = ] zeiger_auf_form_objekt.elements[Index, [SubIndex]]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

```
zeiger_auf_form_objekt      z.B. laut ID-Attribut
```

Eigenschaften:

```
.length      Anzahl der Feldelemente also Feldlänge z.B. bei Collection
```

Methoden:

```
.item()       Referenz auf Feldelement anhand des Integer-Indexes oder des
               Attributnamen (analog zu ID oder NAME-Attribut) liefern
               außer bei Formular mit <INPUT TYPE=image ...>
               da dafür die children-Collection verwendet werden muss !!!
.namedItem()  Referenz auf Eintrag (FeldElement) anhand des Namen
               (analog zu ID oder NAME-Attribut) liefern
.tags()       Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
               siehe tags Collection des DOM
.urns()       Referenz auf Feld aller Elemente mit gemeinsamer URN liefern
```

4.3.2.2.4.2.1.6. window.document.forms Collection des Internet Explorer

Feld der Zeiger aller Formular-Objekte im Dokument
Elementefolge laut HTML-Coding



Syntax:

```
[ var ZeigerAufFeld = ] document.forms
[ var ZeigerAufFeldElement = ] document.forms[Index , [SubIndex]]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.7. window.document.frames Collection des Internet Explorer

Feld der Zeiger aller Frames (beim IE inklusive der inline-Frames, also IFRAMES)

Elementefolge laut HTML-Coding

Element ist eine Referenz auf das Fenster mit Frame-Eigenschaften

Syntax:

Achtung: Für eine Referenz auf das Frame-Objekt ohne diese Collection ist immer document.all zu kodieren.
Bei Fenstererzeugung bitte jedem Fenster seinen eigenen Namen zuweisen (Name nicht doppelt verwenden)

```
[ var ZeigerAufFeld = ] document.frames
[ var ZeigerAufFeldElement = ] document.frames[Index [ ,SubIndex ] ]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

Beispiel für Inhalte zweier Frames tauschen:

Kodierung im Dokument, dass die beiden Frames definiert !

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
                          logischer_name_rahmen2,html_datei2
                          )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch('r1','a.html','r2','b.html')">tauschen</A>
```

Beispiel für Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framennamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument. Argumententrenner sind Doppelpunkte.

Tausch:

```
erster Frame retten und dann mit zweiten Frame überschreiben
zweiten Frame mit drittem Frame überschreiben
.....
vorletzten Frame mit letztem Frame überschreiben
letzten Frame mit geretteten ersten Frame überschreiben
```



```

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)          // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);

                for(var i = 0; i < arguments.length; i++)    // Index ab 0
                {
                    pos_doppelpunkt = arguments[i].indexOf(":");

                    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
                    {
                        // ersten zu ersetzenden Framenamen retten

                        if (i=0)
                        { var rette_logischer_framename =
                            arguments[i].substring(0, pos_doppelpunkt)
                        }

                        // tauschen der logischen Framenamen
                        logischer_framename_zu_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt);

                        logischer_framename_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt + 1);

                        frames[logischer_framename_zu_ersetzender_frame].location.href =
                            logischer_framename_ersetzender_frame;
                    }
                }

                frames[logischer_framename_ersetzender_frame].location.href=
                    rette_logischer_framename;
            }
        }
    }
    // -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
 Attributnamen (analog zu ID oder NAME-Attribut) liefern
 außer bei Formular mit <INPUT TYPE=image ...>
 da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
 (analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.2.1.8. window.document.images Collection des Internet Explorer

Feld der Zeiger aller img Objekte

Elementefolge laut HTML-Koding

Achtung: Per new Image() erzeugte Bilder sind **nicht** Bestandteil der Collection !!

Diese Collection umfasst **nicht** das Objekt input.image, da für dieses Objekt die childrenCollection verwendet werden muss !

Syntax:

```

[ var ZeigerAufFeld = ] document.images
[ var ZeigerAufFeldElement = ] document.images [Index [, SubIndex] ]

```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes



Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.9. window.document.links Collection des Internet Explorer (HTML-Element mit HREF-Attribut)

Feld der Zeiger aller Objekte mit HREF-Eigenschaft (Attribut) sowie aller AREA-Objekte im Dokument, wobei **alle** diese Objekte das Attribut NAME und oder ID besitzen **müssen**, um in dieser Collection referenziert zu sein. Elementefolge laut HTML-Coding

Syntax:

```
[ var ZeigerAufFeld = ] document.links
[ var ZeigerAufFeldElement = ] document.links[Index]
```

Index Integer ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.10. map.areas Collection des Internet Explorer

Zeigerfeld aller area Objekte eines map Objektes (siehe auch dort)

Erzeugung eines Elementes kann nur erfolgen durch die Methoden

.createElement() mit anschliessendem .add()
oder .insertAdjacentHTML()

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_map_objekt.areas
[ var ZeigerAufFeldElement = ] zeiger_auf_map_objekt.areas[Index , SubIndex]
```

zeiger_auf_map_objekt laut ID-Attribut

Index Integer ab 0
oder String z.B. laut ID-Attribut
muss in [] kodiert sein

SubIndex optional
Unterindex also Unterelement eines Elementes
nur kodieren wenn Index ein String ist
Integer

ZeigerAufFeld ist null, wenn keine area Objekte vorhanden

ZeigerAufFeldElement ist null, wenn Feldelement nicht vorhanden

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen
hinzufügen erst nach dem kompletten Laden des Dokumentes
Element erzeugen per Methode .createElement()

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern



.remove() ein Element entfernen aus einer Collection
 .tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
 siehe tags Collection des DOM
 .urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.11. **window.document.select.options** Collection des Internet Explorer

Eine Option ist über dieses Feld der Zeiger aller Optionen anwählbar:

Die Zeiger aller Optionen werden in der Collection document.select.options gesammelt.

Ab IE 5.5 werden option Objekte zusätzlich auch über die Collection **document.all** referenzierbar.

Eine Option aus dem Feld löschen, erfolgt durch Zuweisung des Null-Zeiger (null). Zugleich wird automatisch das Feld kondensiert, also alle Lücken werden entfernt.

Zum Hinzufügen eines option Objektes müssen das select und option Objekt in einem **gemeinsamen** Fenster liegen.

Syntax:

```
[ var ZeigerAufFeld = ] document.zeiger_auf_select_objekt.options
[ var ZeigerAufFeldElement = ] document.zeiger_auf_select_objekt.options [Index [, SubIndex]]
```

Index	Integer ab 0
oder	String Name oder ID des Elementes (analog zu NAME und ID-Attribut)
	muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes
ZeigerAufFeldElement	ist null, wenn Feldelement nicht vorhanden wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection aus diesen Script-Objekten geliefert

zeiger_auf_select_objekt z.B. laut ID-Attribut

Beispiel:

```
var Feld = document.all.tags("SELECT");

if (Feld.length>0)
{
    for (var i=0; i < Feld[0].options.length; i++)
    {
        alert(      "Element " + i
                    + " : mit internem Text = "      + Feld[0].options(i).text
                    + " und angezeigtem Wert = "      + Feld[0].options(i).value);
    }
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen
 hinzufügen erst nach dem kompletten Laden des Dokumentes
 Element erzeugen per Methode .createElement()
 .item() Referenz auf Feldelement anhand des Integer-Indexes oder des
 Attributnamen (analog zu ID oder NAME-Attribut) liefern
 außer bei Formular mit <INPUT TYPE=image ...>
 da dafür die children-Collection verwendet werden muss !!!
 .namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
 (analog zu ID oder NAME-Attribut) liefern
 .remove() ein Element aus einer Collection entfernen
 .tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
 siehe tags Collection des DOM
 .urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.2.1.12. **window.document.selection.controlrange** Collection des Internet Explorer

Feld der Zeiger aller Control-Elemente, die in der **aktiven** Selektion per document.selection Objekt markiert wurden

nur für body Objekt

ein Objekt controlrange existiert nicht

ab IE 5.x

Syntax:

```
[ var ZeigerAufFeld = ] document.selection.controlrange
[ var ZeigerAufFeldElement = ] document.selection.controlrange [Index]
```

Index	Integer ab 0
oder	String Name oder ID des Elementes (analog zu NAME und ID-Attribut)



muss in [] kodiert sein

ZeigerAufFeldElement

ist null, wenn Feldelement nicht vorhanden
wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection
aus diesen Script-Objekten geliefert

Eigenschaften:

.length

Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add()

ein bereits erzeugtes Element einer Collection hinzufügen
hinzufügen erst nach dem kompletten Laden des Dokumentes
Element erzeugen per Methode .createElement()

.execCommand()

Kommando ausführen z.B. im aktuellen Dokument
in aktueller Selektion
im aktuellen Bereich
erst nach dem kompletten Laden des Dokumentes zulässig
Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
Control = Element zur Steuerung analog zum HTML-Element (Tag)
Input-Control = Element mit Eingabeeigenschaft

.item()

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!
.queryCommandEnabled() prüfen ob Kommando ausführbar ist
.queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
.queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird
.queryCommandValue() Wert eines Kommandos liefern
.remove() ein Element aus einer Collection entfernen
.scrollIntoView() Objekt derart scrollen, dass es im Fenster für User sichtbar wird
Objekt muss an sich schon renderbar sein
.select() Selektion eines Textranges (Textbereich, Objekt textrange) oder ControlRange (Control-Elemente)
nur unter Windows 32-Bit
siehe Objekt document.selection
Methoden .createTextRange() .createControlRange() und .createRange()

4.3.2.2.4.2.1.13. window.document.selection.textrange Collection des Internet Explorer

Feld der Zeiger aller textrange Objekte der **aktiven** Selektion

ab IE 5.5

Elternobjekte mit Textrange sind

body Objekt
button Objekt
textarea Objekt
input text Objekt
selection Objekt (nur wenn ein Text selektiert wurde (mit oder ohne HTML))

wobei diese weitere HTML-Elemente enthalten können, die ebenfalls Textbereiche besitzen können

Syntax:

```
[ var ZeigerAufFeld = ] document.selection.textrange
[ var ZeigerAufFeldElement = ] document.selection.textrange[Index [, SubIndex] ]
```

Index Integer ab 0
oder String Name oder ID des Elementes
(analog zu NAME und ID-Attribut)
muss in [] kodiert sein

SubIndex optional
nur kodieren wenn Index ein String ist
Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement

ist null, wenn Feldelement nicht vorhanden
wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection
aus diesen Script-Objekten geliefert

Eigenschaften:

.length

Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem()

Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern



4.3.2.2.4.2.1.14. window.document.TextRange Objekt des Internet Explorer (Textbereich im Dokument)

Objekt des gesamten Plaintextes eines Objektes (Textbereich, Textrange)

basiert auf dem Objekt TextNode

Elternobjekte mit Textrange sind alle Objekte, die eine der nachfolgend beschriebenen Methoden besitzen

z.B. body Objekt
button Objekt
textarea Objekt
input text Objekt
selection Objekt

(nur wenn ein Text selektiert wurde (mit oder ohne HTML))

können weitere HTML-Elemente enthalten, die ebenfalls Textbereiche besitzen können

Textbereich kann in Textteile zerlegt sein: Jeder Teil ist ein Element mit Plaintext.

hat seinen Bereichrahmen (siehe Objekt TextRange.TextRectangle und Collection TextRange.TextRectangle)

**Erzeugung:
in HTML**

innerhalb der Tag-Begrenzer muss Text kodiert sein

z.B. leerer DIV hat keinen Textbereich

kann nachträglich einen Textbereich per Script erhalten

Beispiel: <HTML>
<BODY>
<H1>Hallo</H1>
<CENTER><H2> und willkommen ! <H2></CENTER>
</BODY>
</HTML>

Textrange ist erzeugt worden im BODY-Teil

und enthält nur Plain-Text also

"Hallo und willkommen !"

hat genau einen Zeiger auf die Textrange-Element

Textrange-Element ist z.B. das Wort "Hallo" des H1-Elementes im Container BODY

per Script:

.createTextRange() Textbereich erzeugen

Syntax:

[var Zeiger =] object.createTextRange()

Zeiger auf Range

null wenn TextRange nicht erzeugbar

Beispiel 1:

```
<SCRIPT LANGUAGE="JScript">
var FeldAllerButtonElemente coll = document.all.tags("BUTTON");

if ( (FeldAllerButtonElemente !=null )
    && (FeldAllerButtonElemente.length>0)
    )
{
    var ZeigerAufRange = FeldAllerButtonElemente[0].createTextRange();
    ZeigerAufRange.text = "Clicked";
}
</SCRIPT>
```

Beispiel 2:

```
function ErzeugeUndSelektiereTextRange()
{
    var TextBereich = document.body.createTextRange();
    TextBereich.findText("Testtext");
    TextBereich.select();
}
```

.createRange() Zeiger auf einen Universal-Bereich erzeugen per document.selection Objekt des Dokumentes
Universal-Bereich kann enthalten:

Text (document.selection.textrange Collection
textrange Objekt)

oder Control-Element(e) (document.selection.controlRange Collection)

siehe auch Methoden .createControlRange() .createTextRange() und .createRangeCollection()
Eigenschaft .type

Syntax:

[var Zeiger =] document.selection.createRange()

.createRangeCollection() document.selection.textrange Collection erzeugen per document.selection Objekt des Dokumentes
nur wenn der Browser multiple Selektion unterstützt, so mehr als 1 Feld-Element textrange Objekt
vorhanden bei Mehrfach-Selektion durch User
siehe auch textrange Objekt



Methoden `.createRange()` `.createControlRange()` und `.createTextRange()`

Syntax:

[var Zeiger =] = document.selection.createRangeCollection()

Eigenschaften:

`.boundingHeight` Höhe des Rechteckes in Pixel um den Textbereich des TextRange Objektes
per textrange Objekt

`.boundingLeft` Abstand linker Rand des Rechteckes in Pixel um den Textbereich zur linkem Rand des Container-Objektes,
in dem der Textbereich liegt
per textrange Objekt

`.boundingTop` Abstand oberer Rand des Rechteckes in Pixel um den Textbereich zum oberen Rand des Container-
Objektes,
in dem der Textbereich liegt
per textrange Objekt

`.boundingWidth` Breite des Rechteckes in Pixel um den Textbereich des TextRange Objektes
per textrange Objekt

`.htmlText` HTML-Text im Textbereich
nicht den Plaintext-Anteil liefern
per textrange Objekt
nur unter Windows 32-Bit

`.offsetLeft` X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem
des Elternobjektes (`.offsetParent`)

`.offsetTop` Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem
des Elternobjektes (`.offsetParent`)

`.text` Plain-Text im Textbereich
per textrange Objekt
Dokument muss komplett geladen sein
nur unter Windows 32-Bit

Methoden:

`.collapse()` Textbereich-Zeichen-Zeiger auf Anfang oder Ende des Textbereiches setzten
Hinweis: Zeiger nur auf Plain-Text-Zeichen
Bsp.: `<BODY><P>abc`
Zeiger kann wandern von VOR a bis HINTER c
VOR a entspricht Start des Bereiches
mit Zeigerwert 0
HINTER c entspricht Ende des Bereiches
mit Zeigerwert 3
per textrange Objekt
nur unter Windows 32-Bit

`.compareEndpoints()` Vergleich der Textbereich-Zeichen-Zeiger von 2 Textbereichen
Hinweis: Zeiger nur auf Plain-Text-Zeichen
Bsp.: `<BODY><P>abc`
Zeiger kann wandern von VOR a bis HINTER c
VOR a entspricht Start des Bereiches
mit Zeigerwert 0
HINTER c entspricht Ende des Bereiches
mit Zeigerwert 3
per textrange Objekt
nur unter Windows 32-Bit

`.duplicate()` Zeiger auf eine Duplikat-Instanz eines Textbereiches liefern
per textrange Objekt
nur unter Windows 32-Bit
Prüfung eines Textbereichen auf Kopie eines anderen Textbereiches per Methode `.inRange()`
bzw. `.isEqual()`

`.execCommand()` Kommando ausführen z.B. im aktuellen Dokument
in aktueller Selektion
im aktuellen Bereich
erst nach dem kompletten Laden des Dokumentes zulässig
Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
Control = Element zur Steuerung analog zum HTML-Element (Tag)
Input-Control = Element mit Eingabeeigenschaft

`.expand()` Plain-Text als Teil eines Textbereiches derart ausdehnen, dass er komplett die Dimension des Elternobjektes
(Container-Objektes) einnimmt
per textrange Objekt
nur unter Windows 32-Bit

`.findText()` Text im gesamten Textbereich suchen
per textrange Objekt
nur unter Windows 32-Bit

`.getBookmark()` für Nutzung einer Textmarke siehe Methoden `.getBookmark()` und `.moveToBookmark()`
Textmarke (Bookmark) im Textbereich setzen
Textmarke kann mit Methode `.moveToBookmark()` anpositioniert werden
per textrange Objekt



.getBoundingClientRect()	nur unter Windows 32-Bit Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle prüfen ob 2 Textbereiche sich einschliessen z.B. bei verschachtelten DIV's
.inRange()	per textrange Objekt nur unter Windows 32-Bit
.isEqual()	Auf Identität zweier Textbereiche prüfen per textrange Objekt
.move()	Textbereich-Zeichen-Zeiger bewegen bezüglich aktueller Position im Textbereich per textrange Objekt nur unter Windows 32-Bit
.moveEnd()	Textbereich-Ende neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich per textrange Objekt nur unter Windows 32-Bit
.moveStart()	Textbereich-Anfang neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich per textrange Objekt nur unter Windows 32-Bit
.moveToBookmark()	Textbereich-Zeichen-Zeiger auf eine Textmarke (Bookmark) im Textbereich positionieren Textmarke wurde gesetzt per Methode .getBookmark() per textrange Objekt nur unter Windows 32-Bit
.moveToElementText()	Textbereich in ein Element/Objekt bewegen, das Text enthalten darf per textrange Objekt nur unter Windows 32-Bit
.moveToPoint()	Inhalt des Textbereiches um eine Pixelspanne verschieben (Offset) relativ zur linken oberen Fensterecke nach Verschiebung kann Textbereich leer sein per textrange Objekt nur unter Windows 32-Bit
.parentElement()	Zeiger auf das Elternelement (Container) des Textbereiches liefern Hinweis bei Elementverschachtelung: es wird das direkt um den Textbereich liegende Element referenziert per textrange Objekt nur unter Windows 32-Bit
.pasteHTML()	Textbereich-Inhalt durch Text ersetzen oder leeren Textbereich füllen Text kann auch HTML enthalten Tabelle nur mit kompletten HTML-Code einfügbar, also z.B. keine einzelne Zelle Achtung: Wenn HTML-Code im Text enthalten ist, muss das Elternobjekt auch diesen ansich unterstützen Bsp.: textArea erlaubt kein HTML-Code HTML-Code wird geparkt per textrange Objekt nur unter Windows 32-Bit
.queryCommandEnabled()	prüfen ob Kommando ausführbar ist
.queryCommandIndeterm()	prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState()	Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
.queryCommandSupported()	prüfen ob Kommando im aktuellen Bereich unterstützt wird
.queryCommandValue()	Wert eines Kommandos liefern
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Objekt selektieren z.B. Textbereich: wird hervorgehoben ControlRange: es wird Rechteck-Rahmen erzeugt nur unter Windows 32-Bit
.setEndPoint()	Textbereichanfang bzw. -ende von 2 Textbereichen synchronisieren per textrange Objekt nur unter Windows 32-Bit

4.3.2.2.4.2.1.14.1. *window.document.TextRange.TextRectangle Collection des Internet Explorer*

ab IE 5.x

Feld der TextRectangle-Objekte im HTML-Element (Objekt) mit Plain-Text

TextRectangle-Objekt: Rahmen der Positionierung einer einzelnen Textzeile im Plain-Text
Positionierung bezüglich Koordinatensystem des HTML-Dokumentes

Beispiel: Ein DIV enthält Plain-Text:

Jede im Browser dargestellte Zeile (auch jedes einzelne
) besitzt ein eigenes Rechteck als
Rahmen der Positionierung innerhalb des DIV.

Damit lassen sich Textelemente per Rahmen innerhalb des Dokumentes positionieren.

Achtung: Nach einer Änderung der Fensterdimension (resize) muss die Collection neu erzeugt werden (ist zu programmieren !).



Syntax:

```
[ var FeldZeiger = ] zeiger_auf_textrange.getClientRects();
```

```
[ var FeldElementZeiger = ] FeldZeiger[Index];
```

Index: Integer und ab 0
muss in [] kodiert sein

Zugriff:

FeldZeiger.eigenschaft
FeldZeiger.methode()

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.2.1.14.2. window.document.TextRange.TextRectangle Objekt des Internet Explorer

ab IE 5.x

TextRectangle-Objekt: Rahmen der Positionierung einer einzelnen Textzeile im Plain-Text (Positionierung bezüglich Koordinatensystem des

HTML-Dokumentes)
instanziert als Element der Collection TextRange.TextRectangle

Beispiel: Ein DIV enthält Plain-Text:

Jede im Browser dargestellte Zeile (auch jedes einzelne
) besitzt einen eigenen Rechteck als
Rahmen der Positionierung innerhalb des DIV.

Damit lassen sich Textelemente per Rahmen innerhalb des Dokumentes positionieren.

Achtung: Nach einer Änderung der Fensterdimension (resize) muss die Collection neu erzeugt werden (ist zu programmieren !).

Syntax:

```
[ var Zeiger = ] zeiger_auf_textrectangle.getBoundingClientRect();
```

zeiger_auf_textrectangle ist Element der Collection TextRange.TextRectangle

```
[ var Zeiger = ] zeiger_auf_textrectangle_collection[Index].getBoundingClientRect();
```

Index Integer, ab 0
muss in [] kodiert

Beispiel:

```
<HEAD>
<SCRIPT>
var ZeigerAufTextRectangleCollection;
var Index =0;

function Anzeigen(Zeiger) // Zeiger auf den einzigsten DIV mit Text also mit Textbereich
{
    // aktuelle Collection der Rectangle von Div0 referenzieren:
    // Das ist nötig nach resize des Fensters, da resize leider nicht automatisch erkannt
    // wird !
    ZeigerAufTextRectangleCollection = Zeiger.getClientRects();

    // Anzahl der Elemente ermitteln, also Anzahl der im Browser dargestellte Zeilen,
    // wobei es egal ist, ob diese per <BR> erzeugt wurden oder nicht
    // Hinweis: Zeilenlänge hängt auch von der Fensterbreite ab
    // (automatischer Umbruch)
    // Jede Zeile besitzt ihr eigenes Rectangle
    AnzahlTextRectangle = ZeigerAufTextRectangleCollection.length;

    // aktuellen Index prüfen ob letzte Zeile bereits erreicht wurde
    if (Index > AnzahlTextRectangle -1) // Index ab 1, Anzahl ab 1
    {
        // es wurde die letzte Zeile erreicht

        // Div2 unsichtbar machen also entfärben
        // Hinweis: Div2 liegt auf dem Rectangle von Div0
        ID_Div2.style.display="none";

        // rücksetzen des Index, also mit erster Zeile weitermachen
```



```

        Index = 0;
    }

    // Rechteck der aktuellen Zeile ermitteln unter Beachtung eines eventuellen Scrollens
    var PosRechts    = ZeigerAufTextRectangleCollection [Index].right + ID_Body.scrollLeft;
    var PosLinks     = ZeigerAufTextRectangleCollection [Index].left  + ID_Body.scrollLeft;
    var PosOben1     = ZeigerAufTextRectangleCollection [Index].top   + ID_Body.scrollTop;

    // und Div1 auf die Zeile positionieren, also Zeile einfärben
    ID_Div1.style.top    = PosOben1;
    ID_Div1.style.width  = (PosRechts - PosLinks) - 5;
    ID_Div1.style.display = 'inline';

    // aktuelle Position des Rechteckes von DIV0 ermitteln unter Beachtung eines eventuellen
    // Scrollens
    PosRechts    = Zeiger.getBoundingClientRect().right + ID_Body.scrollLeft;
    PosLinks     = Zeiger.getBoundingClientRect().left  + ID_Body.scrollLeft;
    var PosOben2  = Zeiger.getBoundingClientRect().top   + ID_Body.scrollTop;

    // und Div2 überlagern positionieren
    ID_Div2.style.top    = PosOben2;
    ID_Div2.style.width  = (PosRechts - PosLinks) - 5;
    ID_Div2.style.height = PosOben1 - PosOben2;

    // aber nur einfärben, wenn mindestens 1 Zeile bereits eingefärbt wurde
    if (Index > 0){ ID_Div2.style.display = 'inline'; }

    // Rectangle der nächsten Zeile einstellen
    Index++;
}
</SCRIPT>
</HEAD>
<BODY ID="ID_Body">
    <DIV ID="ID_Div0"
        onclick=" Anzeigen(this)"
    >
        klicke
        <BR>
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
    </DIV>
    <DIV ID="ID_Div1"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:yellow; display:none"
    >
    </DIV>
    <DIV ID="ID_Div2"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:beige; display:none"
    >
    </DIV>
</BODY>

```

Zugriff:

Zeiger.eigenschaft

Eigenschaften:

sind les- und schreibbar

.bottom

untere Pixelposition des Rechteckes um ein Objekt

.left

linke Pixelposition des Rechteckes um ein Objekt

.right

rechte Pixelposition des Rechteckes um ein Objekt

.top

obere Pixelposition des Rechteckes um ein Objekt

Methoden:

keine

4.3.2.2.4.2.2. Verwaltung mehrerer HTML-Elemente gleicher oder verschiedener Arten im HTML-Dokument**4.3.2.2.4.2.2.1. allgemeine HTML-Element bezogene Verwaltung**

Nachfolgend werden Collectionen beschrieben, die der allgemeinen HTML-Element-bezogenen Verwaltung dienen.



4.3.2.2.4.2.2.1.1. *attribute Objekt des Internet Explorer (Attribute-Verwaltung für ein HTML-Element)*

Pseudo-Objekt als Prototyp eines Attributes (einer Eigenschaft) eines HTML-Objektes

Zugriff:**nur über** die Collection attributes als Feld aller Zeiger aller Attribute des Objektes

Beispiel 1:

```

<SCRIPT>
function ShowAttribs(ZeigerAufObjekt)
{
    var ZeigerAufFeld = ZeigerAufObjekt.attributes;

    for (var Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld [Index];

        alert(
            ZeigerAufFeldElement.nodeName
            + '='
            + ZeigerAufFeldElement.nodeValue
            + '('
            + ZeigerAufFeldElement.specified
            + ')'
        );
    }
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function Anzeigen()
{
    for(var Index=0; Index < ID_Liste.attributes.length; Index ++)
    {
        if(ID_Liste.attributes[Index].specified)
        {
            alert(
                ID_Liste.attributes[Index].nodeName
                + " = "
                + ID_Liste.attributes[Index].nodeValue
            );
        }
    }
}
</SCRIPT>
<UL onclick="Anzeigen()">
    <LI ID = "ID_Liste" ACCESSKEY = "L">Listen-Element
</UL>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <SPAN ID="ID_Span" UNSELECTABLE="on">
        Dieser Text ist <B>selektierbar !!<B>
    </SPAN>
</DIV>
</BODY>
</HTML>

```

Beispiel für option objekt eines select objektes:

```

<SCRIPT>
function Anzeigen()
{
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
}
</SCRIPT>

```



```

<SELECT ID="ID_select" onchange = "Anzeigen()">
  <OPTION VALUE="1">Auswahl 1 </OPTION>
  <OPTION VALUE="2">Auswahl 2 </OPTION>
  <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Eigenschaften:

siehe Collection attributes

Methoden:

siehe Collection attributes

4.3.2.2.4.2.2.1.2. attributes Collection des Internet Explorer

Diese Collection sammelt die Zeiger aller Attribute eines HTML-Elementes

Feld der Objekt-Attribute-Referenzen

aber nicht Style-Attribute-Werte des Objektes, da diese in der Eigenschaft .cssText abgeleitet sind (ab IE 5.x)

Feld muss mit den Methoden extra gepflegt werden, da dies nicht automatisch mit der Modifikation von Attributen vollzogen wird.

Feldelement: ist attribute Objekt (siehe dort)

Syntax:

```

[ var FeldZeiger = ] object.attributes
[ var FeldElementZeiger = ] object.attributes[Index]

```

Index: Integer und ab 0
muss in [] kodiert sein

Beispiel 1:

```

<SCRIPT>
function ShowAttribs(ZeigerAufObjekt)
{
    var ZeigerAufFeld = ZeigerAufObjekt.attributes;

    for (var Index = 0; Index < ZeigerAufFeld.length; Index++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld[Index];

        alert(
            ZeigerAufFeldElement.nodeName
            + '='
            + ZeigerAufFeldElement.nodeValue
            + ' ('
            + ZeigerAufFeldElement.specified
            + ')';
    }
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function Anzeigen()
{
    for(var Index=0; Index < ID_Liste.attributes.length; Index++)
    {
        if(ID_Liste.attributes[Index].specified)
        {
            alert(
                ID_Liste.attributes[Index].nodeName
                + " = "
                + ID_Liste.attributes[Index].nodeValue
            );
        }
    }
}
</SCRIPT>
<UL onclick="Anzeigen()">
  <LI ID = "ID_Liste" ACCESSKEY = "L">Listen-Element
</UL>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>

```




```

        <SPAN ID="ID_Span" UNSELECTABLE="on">
            Dieser Text ist <B>selektierbar !!<B>
        </SPAN>
    </DIV>
</BODY>
</HTML>

```

Beispiel für option objekt eines select objektes:

```

<SCRIPT>
    function Anzeigen()
    {
        var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
        alert(Kette);
    }
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
    <OPTION VALUE="1">Auswahl 1 </OPTION>
    <OPTION VALUE="2">Auswahl 2 </OPTION>
    <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Eigenschaften:

.expando Wirksamkeit von Attributen ein/aus
 true ein, Default
 false aus

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <SPAN ID="ID_Span" UNSELECTABLE="on">
        Dieser Text ist <B>selektierbar !!<B>
    </SPAN>
</DIV>
</BODY>
</HTML>

```

.length Anzahl der Feldelemente also Feldlänge
 .specified prüfen ob Objekt Attribute hat
 true, so Attribute ist vorhanden
 false, so keine Attribute vorhanden

Beispiel:

```

<SCRIPT>
    function Anzeigen()
    {
        var FeldAllerAttribute = ID_List.attributes;
        alert(FeldAllerAttribute(0).nodeName);      // Knotenname

        for( var i=0; i< FeldAllerAttribute.length; i++)
        {
            // neues LI-Element als Knoten der Liste anhängen
            var NeuesListenElement =document.createElement("LI");
            ID_List.appendChild(NeuesListenElement);

            // Wert des neuen LI-Elementes ist Text, also Textknoten
            var WertNeuesListenElement = document.createTextNode(
                i
                + " "
                + FeldAllerAttribute (i).nodeName
                + " = "
                + FeldAllerAttribute (i).nodeValue
            );

            NeuesListenElement.appendChild(WertNeuesListenElement);

            // auf specified prüfen
            if(FeldAllerAttribute (i).nodeValue != null )
            {
                alert(      FeldAllerAttribute(i).nodeName

```



```

        + " specified: "
        + FeldAllerAttribute(i).specified // boolean
    );
    }
}
}
</SCRIPT>
<UL ID="ID_List" onclick = " Anzeigen()">
    <LI>Klick mich
</UL>

```

.value Wert eines Objekt-Attributes
Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar

Beispiel

```

<SCRIPT>
function Anzeigen()
{
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
}
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
    <OPTION VALUE="1">Auswahl 1 </OPTION>
    <OPTION VALUE="2">Auswahl 2 </OPTION>
    <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Methoden:

.getNamedItem() Zeiger auf Attribut liefern anhand des Attributnamen (analog zu ID oder NAME-Attribut)
ab IE 6.x

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function Init()
{
    var ZeigerAufFeld = ID_P.attributes;
    var ZeigerAufFeldElement = ZeigerAufFeld.getNamedItem("align");
    alert("ALIGN Attribut Wert = " + ZeigerAufFeldElement.value);
}
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P" ALIGN="center">Test</P>
</BODY>
</HTML>

```

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
var ZeigerAufCollectionDocumentAll = document.all;

if (ZeigerAufObjekt!=null)
{
    for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
    { alert(ZeigerAufCollectionDocumentAll.item(i).tagName); }
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
function Init()
{
    var ZeigerAufFeld = ID_P.attributes;
    for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
        // hier numerischer Index
        var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
    }
}

```



```

// true oder false
var KnotenName = ZeigerAufFeldElement.nodeName;
// String
var KnotenWert = ZeigerAufFeldElement.nodeValue;
alert(
    "Knotenname = "
    + KnotenName
    + " mit spezifiziert = "
    + AttributWertSpezifiziert
    + " und Wert = "
    + KnotenWert
);
}
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>
.removeNamedItem()    Attribut entfernen anhand Attributname (analog zu ID oder NAME-Attribut),
                        wobei danach der Standard-Attributwert automatisch weiterverwendet wird
                        (falls Standard vorhanden ist),
                        und Zeiger auf gelöscht Attribut liefern
                        ab IE 6

```

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Entfernen()
    {
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.removeNamedItem("TITLE");
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV onclick="Entfernen();" ID="ID_Div" TITLE="Tooltip-Text ">
        Klick um den Tooltip-Text zu entfernen
    </DIV>
</BODY>
</HTML>
.setNamedItem()        Attribut hinzufügen anhand Zeiger auf Attribut
                        wenn noch nicht im Feld vorhanden, so Anhängen an das Feldende
                        wenn schon im Feld vorhanden, so überschreiben und Referenz auf das
                        überschriebene Attribut (vor dem Überschreiben) liefern
                        Bsp: Attribut erzeugen document.createAttribute("title");
                        Hinweis: in HTML können Attributnamen groß oder klein geschrieben werden
                        ab IE 6

```

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Hinzufuegen()
    {
        // Attribut mit Wert erzeugen
        var ZeigerAufAttribut = document.createAttribute("title");
        ZeigerAufAttribut.value = "Tooltip-Text ";

        // Attribut als Feldelement anhängen
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
    }
</SCRIPT>
</HEAD>
<BODY onload="Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

4.3.2.2.4.2.2.1.3. childNodes Collection des Internet Explorer

Feld der Zeiger aller Kinder-Knoten eines Objektes, also Feld der Zeiger aller HTML-Knoten-Kinder **und** Textknoten-Kinder des Objektes
Collection dient zur Ermittlung **nur von Knoteneigenschaften laut DOM**, falls diese im Element implementiert sind:



Elementefolge NICHT laut HTML-Coding sondern laut DOM.

Element (Knoten) kann per HTML oder Methode `.createElement()` erzeugt worden sein (falls Methode erlaubt ist).
Diese Collection sammelt die Zeiger aller Kinder**knoten** eines HTML-Elementes.

Syntax:

```
[ var ZeigerAufFeld = ] object.childNodes
[ var ZeigerAufFeldElement = ] object.childNodes[Index]

object                Zeiger auf Elternobjekt

Index                Integer und ab 0
                    muss in [ ] kodiert sein

ZeigerAufFeldElement    ist null, wenn Feldelement nicht vorhanden
```

Zugriff auf Element:

Je nach Art des Kind-Elementes stehen dem Kind Eigenschaften und Methoden zur Verfügung (siehe Objektbeschreibungen):

```
object.childNodes[Index].eigenschaft_des_kindes
object.childNodes[Index].methode_des_kindes
```

```
object    Zeiger auf Elternobjekt

Index    Integer und ab 0
        muss in [ ] kodiert sein
```

Beispiel 1:

```
<SCRIPT>
    var ZeigerAufFeld = ID_Body.childNodes;
</SCRIPT>
<BODY ID="ID_Body">
    <SPAN>Test </SPAN>
</BODY>
```

Beispiel 2:

```
// DIV erzeugen
var ZeigerAufDivKnoten = document.createElement("DIV");

// B-Tag im DIV erzeugen
var ZeigerAufBKnoten = document.createElement("B");
ZeigerAufDivKnoten.insertBefore(ZeigerAufBKnoten);

// erst jetzt das DIV in den BODY einfügen, also DIV sichtbar machen
document.body.insertBefore(ZeigerAufDivKnoten);

// Collection referenzieren
var ZeigerAufFeld = ZeigerAufDivKnoten.childNodes;
```

Beispiel 3:

```
<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPTag = document.all.tags("P");
    if (ZeigerAufFeldAllerPTag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
        { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline"; } // auch .item(i) kodierbar
    }
</SCRIPT>
```

Beispiel 4:

```
<SCRIPT>
    var ErstesKind_Index = 0;        // Index ab 0
    var ErstesKind_Name = Liste.childNodes(ErstesKind_Index).nodeName;
    alert(ErstesKind_Name);          // liefert den Tagnamen 'LI' von Listenelement 1
                                     // Hinweis: Listenelement 1 ist der Wert des Kindes

</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listenelement 2
        <LI> Listenelement 3
    </UL>
</BODY>
```



Beispiel 5:

```

<SCRIPT>
function KnotenWertAendern( ZeigerAufListe,
                           IndexVonListenElement,    // immer ab 0
                           Zeichenkette               // Listenlement muss Text sein
                           )
{
    var ReturnWert=false;    // Annahme: Änderung schlägt fehl

    // prüfen auf UL-Tag
    if (ZeigerAufListe.nodeName == "UL")
    {
        // Anzahl der Listenelemente holen
        var AnzahlListenelemente= ZeigerAufListe.childNodes.length;
                                                // immer ab 1

        // und Anzahl und Index prüfen
        if ( (AnzahlListenelemente > 0)    // immer ab 1
            && (IndexVonListenElement >= 0) // immer ab 0
            && (IndexVonListenElement < AnzahlListenelemente)
                                                // Index ist zulässig zur Anzahl
        )
        {
            // Zeiger auf das Listenelement laut Index holen
            var ZeigerAufListenElement =
                ZeigerAufListe.childNodes[IndexVonListenElement];

            // existiert das Listenelement ?
            if (ZeigerAufListenElement)
            {
                // ZeigerAufListenElement ist nicht null

                // Listenelement ist Textelement ?
                if (ZeigerAufListenElement.nodeType == 3)
                {
                    ZeigerAufListenElement.nodeValue =
                        Zeichenkette;
                    ReturnWert =true;
                }
            }
        }
    }

    return ReturnWert;
}
</SCRIPT>
<UL ID="Liste" onclick=" KnotenWertAendern(this, 0, 'Listenelement Neu')">
    <LI>Listenelement alt
</UL>

```

Beispiel 6:

```

<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>

```

Beispiel 7:

```

<SCRIPT>
function Anzeige()
{
    var ZeigerAufOnClickEventQuelle=event.srcElement;
    var Feld =
        ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
    alert(
        "Anzahl LI : "
        + Feld.length
    );
}

```



```

        + "\nErster Eintrag: "
        + Feld [0].childNodes[0].nodeValue
    );
}
</SCRIPT>
<UL onclick="Anzeige()">
    <LI>Menuepunkt 1
    <UL>
        <LI> Menuepunkt 1.1
        <OL>
            <LI> Menuepunkt 1 1.1
            <LI> Menuepunkt 1 1.2
        </OL>
        <LI> Menuepunkt 1.2
        <LI> Menuepunkt 1.3
    </UL>
    <LI> Menuepunkt 2
    <UL>
        <LI> Menuepunkt 2.1
        <LI> Menuepunkt 2.3
    </UL>
    <LI> Menuepunkt 3
</UL>

```

Eigenschaften:**.length**

Anzahl der Feldelemente also Feldlänge

Methoden:**.item()**

Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>

```



```

</HTML>
.urns()           Referenz auf Feld aller Elemente mit gemeinsamer URN liefern
Beispiel:
<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
    var Text = "";

    if (ZeigerAufFeldAllerURN1 != null)
    {
        for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
        {Text += ZeigerAufFeldAllerURN1.item(i).id + ' ';}
        alert (Text);
    }
</SCRIPT>

```

4.3.2.2.4.2.1.4. children Collection des Internet Explorer

Feld der Zeiger aller **HTML-Elemente-Kinder** eines Objektes

Collection dient **auch** zur Ermittlung von Knoteneigenschaften, wenn diese im Element implementiert sind.

Elementefolge laut HTML-Coding und nicht laut DOM

Element kann per HTML oder Methode .createElement() erzeugt worden sein (falls Methode erlaubt ist).

Für das Objekt form.input image **muss** die children Collection verwendet werden.

Syntax:

```

[ var ZeigerAufFeld = ] object.children
[ var ZeigerAufFeldElement = ] object.children[Index [, SubIndex] ]

```

object		Zeiger auf Elternobjekt
Index	oder	Integer ab 0 String Name oder ID des Elementes laut ID-Attribut bzw. NAME-Attribut muss in [] kodiert sein
SubIndex		optional Integer Unterindex also Unterelement eines Elementes nur kodieren wenn Index ein String ist
ZeigerAufFeldElement		ist null, wenn Feldelement nicht vorhanden

Beispiel 1:

```

<SCRIPT>
    var ZeigerAufFeld = ID_Body.children;
</SCRIPT>
<BODY ID="ID_Body">
    <SPAN >Test </SPAN>
</BODY>

```

Beispiel 2:

```

<SCRIPT>
    function Loeschen()
    {
        ID_Span.children[0].clearAttributes();
    }
</SCRIPT>
<SPAN ID="ID_Span">
    <DIV ID="ID_Div"
        ATTRIBUTE1="true"
        ATTRIBUTE2="true"
        onclick="alert('click');"
        onmouseover="this.style.color='#0000FF';"
        onmouseout="this.style.color='#000000';"
    >
        Test eines<B>Div</B>Elementes.
    </DIV>
</SPAN>
<INPUT TYPE="button" VALUE=" Loeschen" onclick="Loeschen()">

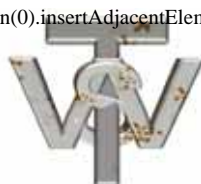
```

Beispiel 3:

```

<SCRIPT>
    function Hinzufuegen()
    {
        var ZeigerAufLI = document.createElement("LI");
        ID_Liste.children(0).insertAdjacentElement("beforeBegin", ZeigerAufLI);
    }

```



```

        ZeigerAufLI.innerText = "Listeneintrag 0";
    }
</SCRIPT>
<BODY>
    <OL ID = "ID_Liste">
        <LI>Listeneintrag 1</LI>
        <LI>Listeneintrag 2</LI>
        <LI>Listeneintrag 3</LI>
    </OL>
    <INPUT TYPE = "button" VALUE = "Hinzufuegen" onclick="Hinzufuegen()">
</BODY>

```

Beispiel 4:

```

<SCRIPT>
    function Tauschen()
    {
        Liste.children(0).swapNode(Liste.children(1));
    }
</SCRIPT>
<UL ID = Liste>
    <LI>Listeneintrag 1
    <LI>Listeneintrag 2
    <LI>Listeneintrag 3
    <LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Tauschen" onclick = "Tauschen()">

```

Beispiel 5:

```

<HEAD>
<SCRIPT>
    function Entfernen()
    {
        // versuche den Text zu entfernen
        try
        {
            var KindZeigerAufTextImDiv = ID_Div.children(0);
            ID_Div.removeChild(KindZeigerAufTextImDiv);
            // Achtung: Der Text ist noch sichtbar !!!!
        }
        // oder fange das Ereignis des bereits entfernten Textes ein
        // und behandle das Ereignis
        catch(x)
        {
            alert(
                "Text wurde entfernt !\n"
                + "Das Dokument muss neu geladen werden !
            );
            document.location.reload();
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick="Entfernen()">
        Klick, um diesen Text zu entfernen !
    </DIV>
</BODY>

```

Beispiel 6:

```

<HEAD>
<SCRIPT>
    function Ersetze()
    {
        var KindZeigerAufDivText = ID_Div.children(0);
        var RetteInnerHTML = KindZeigerAufDivText.innerHTML;

        // prüfen auf Tag im Div-Text
        if (KindZeigerAufDivText.tagName=="B")
        {
            // Bold-Tag <B>gefunden, also I-Tag erzeugen
            var ZeigerAufNeuenSchriftStilTag =document.createElement("I");

            // komplettes ersetzen von Div-Text,
            ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
        }
    }

```




```

        ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
    else
    {
        // keinen Bold-Tag <B>gefunden
        var ZeigerAufNeuenSchriftStilTag =document.createElement("B");

        // komplettes ersetzen von Div-Text,
        ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
        ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick="Ersetze()">
        Klicke für den Wechseln des <B>Schriftstils<B>
    </DIV>
</BODY>

```

Eigenschaften:**.length**

Anzahl der Feldelemente also Feldlänge

Methoden:**.item()**

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>

```

.tags()

Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

Beispiel:



```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPTag = document.all.tags("P");
    if (ZeigerAufFeldAllerPTag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
        { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
    }
</SCRIPT>

```

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
    var Text = "";

    if (ZeigerAufFeldAllerURN1 != null)
    {
        for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
        {Text += ZeigerAufFeldAllerURN1.item(i).id + ' ';}
        alert (Text);
    }
</SCRIPT>

```

4.3.2.2.4.2.2.1.5. tags Collection des Internet Explorer

Diese Collection sammelt die Zeiger aller HTML-Elemente, die gemeinsamen HTML-Tag besitzen. wird **nur** von Collectionen und Objekten instanziiert, die die Methode .tags() besitzen

z.B. childNodes Collection des DOM
 children Collection des DOM
 document.all Collection des DOM

Zugriff auf das Feld **nur** über die Methode .tags()

Syntax:

[var Zeiger =] zeiger_auf_collection_oder_objekt.tags(Kette)

Kette String mit HTML-Tag-Bezeichner

fZeiger weist auf ein leeres Feld, wenn keine HTML-Elemente mit dem Tag gefunden wurden

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPTag = document.all.tags("P");
    if (ZeigerAufFeldAllerPTag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
        { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
    }
</SCRIPT>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

keine

4.3.2.2.4.2.2.2. spezielle HTML-Element bezogene Verwaltung

4.3.2.2.4.2.2.2.1. Erzeugung ausgewählter eines HTML-Elemente durch Makro (command Objekt des Internet Explorer)

Objekt zum Verwalten von externen vordefinierten Kommandos des IE. Diese Kommandos sind eine andere Variante von Objekterzeugungen (z.B. von HTML-Elementen) als Analogon zu Makros nutzen das DOM

Das Objekt besitzt nur Methoden, die an andere Objekte vererbt werden.

Die meisten Methoden sind **erst** nach dem kompletten Laden des Dokumentes anwendbar.

Falls die Methoden Werte liefern, so sind die Werttypen kommandospezifisch.

Beispiel 1:

```

<HTML>
<BODY>
    <H1 UNSELECTABLE="on">Demo</H1>
    <SCRIPT>
        function AddLink()
        {
            var SelektierterText = document.selection.createRange();

            if (!SelektierterText == "")
            {

```



```

// Link erzeugen
document.execCommand("CreateLink");

if (SelektierterText.parentElement().tagName == "A")
{
    // markierten Text mit Eltern-Url ersetzen
    SelektierterText.parentElement().innerText=
        SelektierterText.parentElement().href;

    // Vordergrundfarbe setzen im Dokument
    document.execCommand(
        "ForeColor","false","#FF0033");
}
}
else
{alert("Bitte im blauen Text selektieren !");
}
}
</SCRIPT>
<P UNSELECTABLE="on">
    Selektiere (markiere) im nachfolgenden blauen Text die Stelle mit
    dem Text MARKIERE_MICH.<BR>
    Danach auf das Button klicken.<BR>
    Anstelle von MARKIERE_MICH im blauen Text erscheint dort
    nun eine Url.
</P>
<P STYLE="color=#3366CC">
    Meine beliebteste Webseite MARKIERE_MICH bitte besuchen !
</P>
<BUTTON onclick="AddLink()" UNSELECTABLE="on">Klick</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function HandlerFuerOnMoveStart()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "green";
        ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
    }

    function HandlerFuerOnMove ()
    {
        ID_Span1.innerHTML = event.srcElement.offsetLeft;
        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandlerFuerOnMoveEnd()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    // 2-D Positionierung einschalten
    document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
    onmove="HandlerFuerOnMove();"
    onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>

```



```

offserTop = <SPAN ID="ID_Span2"></SPAN>
<BR>
<DIV CONTENTEDITABLE="true">
  <DIV STYLE=
    "position:absolute;width:300px;height:100px; background-color:red;"
  >
    bewegbarer DIV
  </DIV>
</DIV>
</BODY>
</HTML>

```

Eigenschaften:

keine

Methoden:`.execCommand()`

Kommando ausführen z.B. im aktuellen Dokument

in aktueller Selektion

im aktuellen Bereich

erst nach dem kompletten Laden des Dokumentes zulässig

Hinweis: Selektion = Markierung z.B. von Textbereich (Block)

Control = Element zur Steuerung analog zum HTML-Element (Tag)

Input-Control = Element mit Eingabeeigenschaft

Syntax:

```
var Wert = object.execCommand(Command [, UserInterface] [, Value])
```

Command String als Kommando, wobei Gross-Kleinschreibung egal ist

"2D-Position"	absolute Positionierung von Elementen durch Bewegen im Dragging erlauben
"AbsolutePosition"	Element-Eigenschaft der Position auf "absolute" setzen nur für selektierbare Elemente nicht für STYLE-Deklarationen im Dokument
"BackColor"	lesen oder setzen der Hintergrundfarbe der aktuellen Selektion
"Bold"	Wechsel zwischen bold und nonbold in der aktuellen Selektion
"Copy"	Aktuelle Selektion in das Clipboard kopieren
"CreateBookmark"	Bookmark setzen oder lesen für aktuelle Selektion bzw. Einfügepunkt
"CreateLink"	Hyperlink in der aktuellen Selektion setzen oder einfügen bei Einfügen erscheint Dialog-Box
"Cut"	Aktuelle Selektion in das Clipboard verschieben
"Delete"	Aktuelle Selektion löschen Hinweis: Dokument nicht löschar
"FontName"	Font für aktuelle Selektion setzen oder holen
"FontSize"	Fontsize für aktuelle Selektion setzen oder holen
"ForeColor"	Vordergrundfarbe (Textfarbe) für aktuelle Selektion setzen oder holen
"FormatBlock"	Blockformat setzen
"Indent"	Ident des selektierten Textes erhöhen
"InsertButton"	in Textselektion das Button-Control einfügen, wenn eines bereits vorhanden so überschreiben
"InsertFieldset"	in Textselektion die Box einfügen, wenn eine bereits vorhanden so überschreiben
"InsertHorizontalRule"	in Textselektion die horizontale Linie einfügen wenn eine bereits vorhanden so überschreiben
"InsertIFrame"	in Textselektion den inline-frame (IFRAME) einfügen wenn einer bereits vorhanden so überschreiben
"InsertImage"	in Textselektion das Image einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputButton"	in Textselektion das Input-Button-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputCheckbox"	in Textselektion das Input-Check-Box-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputFileUpload"	in Textselektion das Input-File-Upload-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputHidden"	in Textselektion das Input-Hidden-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputImage"	in Textselektion das Input-Image-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputPassword"	



		in Textselektion das Input-Password-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputRadio"		in Textselektion das Input-Radio-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputReset"		in Textselektion das Input-Reset-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputSubmit"		in Textselektion das Input-Submit-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputText"		in Textselektion das Input-Text-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertMarquee"		in Textselektion das Marquee einfügen wenn eines bereits vorhanden so überschreiben
"InsertOrderedList"		Wechsel zwischen ordered list und normalen Block für aktuelle Textselektion
"InsertParagraph"		in Textselektion Zeilenumbruch einfügen wenn eines bereits vorhanden so überschreiben
"InsertSelectDropdown"		in Textselektion das Drop-Down-Selections-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertSelectListbox"		in Textselektion die List-Box-Selektion einfügen wenn eines bereits vorhanden so überschreiben
"InsertTextArea"		in Textselektion das Input-Textarea-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertUnorderedList"		Wechsel zwischen unordered list und normalen Block für aktuelle Textselektion
"Italic"		Wechsel zwischen italic und non-italic für aktuelle Textselektion
"JustifyCenter"		zentrieren für aktuelle Textselektion
"JustifyLeft"		linksbündig für aktuelle Textselektion
"JustifyRight"		rechtssbündig für aktuelle Textselektion
"MultipleSelection"		Mehrfachselektion per CTRL+ bzw. Shift + erlauben
"Outdent"		Outdent des selektierten Textes erniedrigen
"OverWrite"		Wechsel zwischen überschreiben und nicht überschreiben
"Paste"		Aktuelle Selektion aus Clipboard überschreiben
"Print"		Druck-Dialogbox öffnen
"Refresh"		aktuelles Dokument refreshen
"RemoveFormat"		formatierende Tags der aktuellen Selektion entfernen
"SaveAs"		Aktuelles Dokument speichern als Datei
"SelectAll"		alles markieren (selektieren)
"UnBookmark"		Alle Bookmark der aktuellen Selektion entfernen
"Underline"		Wechsel zwischen underline und nicht-underline für aktuelle Textselektion
"Unlink"		Alle Hyperlink der aktuellen Selektion entfernen
"Unselect"		Alles demarkieren (de-selektieren)
UserInterface	false	Default user interface soll nicht angezeigt werden (Dialogbox)
	true	user interface soll angezeigt werden (falls Kommando das unterstützt)
Value	z.B. String, number immer passend zu Command, kann optional sein Kodierung null (nicht numerisch Null !!) als Wert entspricht Weglassen des optionalen Wertes	

Kommando	IE ab	Dialogbox	Value
2D-Position	5.5	nein	true für on false für off
AbsolutePosition	5.5	nein	true für "absolut" false für nicht "absolut"
BackColor	4.x	nein	rrggbb OHNE führendes # vordefinierter Farbname (browserspezifisch)
Bold	4.x	nein	null oder omit oder weglassen
Copy	4.x	nein	null oder omit oder weglassen
CreateBookmark	4.x	nein	String mit Ankername keine Leerkette
CreateLink	4.x	ja	String mit Url



Cut	4.x	nein	keine Leerkette
Delete	4.x	nein	null oder omit oder weglassen
FontName	4.x	nein	null oder omit oder weglassen
			String mit Fontname
			oder Fontliste
			Fontliste: Folge von
			Fonteigenschaften
FontSize	4.x	nein	String mit Fontsize von
			einschliesslich 1 bis
			einschliesslich 7
ForeColor	4.x	nein	rrggbb OHNE führendes #
			vordefinierter Farbname
			(browserspezifisch)
FormatBlock	4.x	nein	String mit Block-Tag
Indent	4.x	nein	null oder omit oder weglassen
InsertButton	4.x	nein	String mit Attributen
			des Button-Control
InsertFieldset	4.x	nein	String mit Attributen der Box
InsertHorizontalRule	4.x	nein	String mit Attributen der Linie
InsertIFrame	4.x	nein	String mit Attributen des
			IFRAME
InsertImage	5.x	ja	String mit Pfad und Dateiname
InsertInputButton	4.x	nein	String mit Attributen des Input-
			Button-Control
InsertInputCheckbox	4.x	nein	String mit Attributen des Input-
			Checkbox-Control
InsertInputFileUpload	4.x	nein !!!	String mit Attributen des Input-
			Fileupload-Control
InsertInputHidden	4.x	nein	String mit Attributen des Input-
			Hidden-Control
InsertInputImage	4.x	nein	String mit Attributen des Input-
			Image-Control
InsertInputPassword	4.x	nein	String mit Attributen des Input-
			Password-Control
InsertInputRadio	4.x	nein	String mit Attributen des Input-
			Radio-Control
Kommando	IE ab	Dialogbox Value	
InsertInputReset	4.x	nein	String mit Attributen des Input-
			Reset-Control
InsertInputSubmit	4.x	nein	String mit Attributen des Input-
			Submit-Control
InsertInputText	4.x	nein	String mit Attributen des Input-
			Text-Control
InsertMarquee	4.x	nein	String mit Attributen des
			Marquee-Control
InsertOrderedList	4.x	nein	String mit Attributen des
			Ordered-List-Control
InsertParagraph	4.x	nein	String mit Attributen des
			Paragraph-Control
InsertSelectDropdown	4.x	nein	String mit Attributen des Drop-
			Down-Control
InsertSelectListbox	4.x	nein	String mit Attributen des List-
			Box-Control
InsertTextArea	4.x	nein	String mit Attributen des Text-
			Area-Control
InsertUnorderedList	4.x	nein	String mit Attributen des
			Unordered-List-Control
Italic	4.x	nein	null oder omit oder weglassen
JustifyCenter	4.x	nein	null oder omit oder weglassen
JustifyLeft	4.x	nein	null oder omit oder weglassen
JustifyRight	4.x	nein	null oder omit oder weglassen
MultipleSelection	5.5	nein	true für on
			false für off
Outdent	4.x	nein	null oder omit oder weglassen
OverWrite	4.x	nein	true für Überschreiben
			false für Nicht-Überschreiben
Paste	4.x	nein	null oder omit oder weglassen
Print	5.5	ja	null oder omit oder weglassen
			kein String !!!
Refresh	4.x	nein	null oder omit oder weglassen



RemoveFormat	4.x	nein	null oder omit oder weglassen
SaveAs	4.x	ja	wenn Dialogbox anzeigen so kann Wert null sein wenn Dialogbox nicht anzeigen so muss Wert die Paramter enthalten
SelectAll	4.x	nein	null oder omit oder weglassen
UnBookmark	4.x	nein	null oder omit oder weglassen
Underline	4.x	nein	null oder omit oder weglassen
Unlink	4.x	nein	null oder omit oder weglassen
Unselect	4.x	nein	null oder omit oder weglassen
Wert	true wenn Kommando ausgeführt wurde false wenn Kommando nicht ausgeführt wurde		

.queryCommandEnabled() prüfen ob Kommando ausführbar ist
 .queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht
 .queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
 .queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird
 .queryCommandValue() Wert eines Kommandos liefern

4.3.2.2.4.2.2.2.2. Erzeugung eines privaten HTML-Elementes (custom Objekt des Internet Explorer)

Objekt, das ein durch den Programmierer (Customer) frei definiertes HTML-Element (Customer-Tag) repräsentiert
 auf einem XML-Namensraum (XMLNS) basiert
nur verwendet wird für
 Behavior-Definition per *.htc-Datei (nicht Standard-Behavior des IE)
globale Style-Definition im HEAD

ab IE 5.x

siehe Objekt style

Syntax:

```
<HTML namens_raum_liste>
<HEAD>
  <STYLE>
    @media all {
      namen_raum_listen_element_1\:freier_style_name_1 { liste_1 }
      .....
      namen_raum_listen_element_n\:freier_style_name_n { liste_n }
    }
  </STYLE>
</HEAD>
<BODY>

</BODY>
</HTML>
```

<HTML namens_raum_liste> mit Leerzeichen getrennte Elemente

Element mit Aufbau

XMLNS:freier_tag [= wert]

XMLNS: festkodiert

freier_tag wie normaler HTML-Tag aber frei definiert
 Empfehlung: Gross-Schreibung

wert von freier_tag
 optional
 String

@media all { } Pflichtkodierung

namens_raum_liste_element Der Doppelpunkt muss entwertet werden, also \: kodieren, da dieser bereits ein
 Trenner laut Syntax für Style-Eigenschaften in liste ist.

freier_style_name Bezeichner eines globalen Style **zum** Customer-Tag
 Empfehlung: Gross-Schreibung

liste per Semikolon getrennte Style-Eigenschaften wie beim
 STYLE-Attribut, also den Trenner Doppelpunkt
nicht entwerten !



Kodierung innerhalb BODY:

```
<freier_tag:freier_style_name>
.....
</freier_tag:freier_style_name>
```

Beispiel 1:

```
<HTML XMLNS:MY1
      XMLNS:MY2="www.test.de"
>
.....
```

Beispiel 2:

```
<HTML XMLNS:MY>
<HEAD>
<STYLE>
    @media all {
        MY\:JUSTIFY { text-align:justify; width:100 }
    }
</STYLE>
</HEAD>
<BODY>
<MY:JUSTIFY>
    Text mit Style laut userdefiniertem MY:JUSTIFY-Tag
</MY: JUSTIFY>
</BODY>
</HTML>
```

Beispiel 3:

```
<HTML XMLNS:MY1
      XMLNS:MY2
      XMLNS:MY3
>
<HEAD>
<STYLE>
    @media all {
        MY1\;TEXT_FARBE_ROT    { color: red; }
        MY2\;TEXT_FARBE_GRUEN { color: green; }
        MY3\;TEXT_FARBE_BLAU   { color: blue; }
    }
</STYLE>
</HEAD>
<BODY>
    <MY1:TEXT_FARBE_ROT>
        Text1
    </MY1:TEXT_FARBE_ROT>

    <MY2:TEXT_FARBE_GRUEN>
        Text2
    </MY2:TEXT_FARBE_GRUEN>

    <MY3:TEXT_FARBE_BLAU>
        Text3
    </MY3:TEXT_FARBE_BLAU>
</BODY>
</HTML>
```

Hinweis: Da es sich in allen Styles **immer** um eine Textfarbe handelt und sich die freien Style-Bezeichner unterscheiden, hätte auch kodiert werden können:

```
<HTML XMLNS:MY>
....
    @media all {
        MY:TEXT_FARBE_ROT    { color: red; }
        MY:TEXT_FARBE_GRUEN { color: green; }
        MY:TEXT_FARBE_BLAU   { color: blue; }
    }
....
<BODY>
```




```

<MY:TEXT_FARBE_ROT>
    Text1
</MY:TEXT_FARBE_ROT>

<MY:TEXT_FARBE_GRUEN>
    Text2
</MY:TEXT_FARBE_GRUEN>

<MY:TEXT_FARBE_BLAU>
    Text3
</MY:TEXT_FARBE_BLAU>
</BODY>

```

Beispiel 4:

```

<HTML XMLNS:MY >
<HEAD>
<STYLE>
    @media all {
        MY\;HTC_DATEI { behavior: url(test.htc); }
    }
</STYLE>
</HEAD>
<BODY>
    <MY:HTC_DATEI>
        ....
    </MY:HTC_DATEI>
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.blockDirection	Umfluss um ein Objekt "ltr" Umfluss von links nach rechts "rtl" Umfluss von rechts nach links
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.hideFocus	Focussierbarkeit true Focus ist nicht möglich false Default Focus ist möglich
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,



	also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler



	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernenbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Klick auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein ! "scrollbarDown" oder "down" Down scroll Pfeil "scrollbarHThumb" horizontaler Scrollbalken "scrollbarLeft" oder "left" Left scroll Pfeil "scrollbarPageDown" oder "pageDown" Page-down Scrollbalken "scrollbarPageLeft" oder "pageLeft" Page-left Scrollbalken "scrollbarPageRight" oder "pageRight" Page-right Scrollbalken "scrollbarPageUp" oder "pageUp" Page-up Scrollbalken "scrollbarRight" oder "right" Right scroll Pfeil "scrollbarUp" oder "up" Up scroll Pfeil "scrollbarVThumb" vertikaler Scrollbalken
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden



	expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert

4.3.2.2.4.2.3. Cookie-Verwaltung im HTML-Dokument (document.cookie Collection des Internet Explorer)

4.3.2.2.4.2.3.1. Zweck von Cookies

Cookies dienen dem Zwischenspeichern von Daten auf der Festplatte des Client (Users)
auch über beliebig viele Online-Sitzungen hinaus
auch permanent endlos (Cookie ohne Verfallsdatum)



können von einem oder mehreren Servern geschrieben und gelesen werden auf/von der Festplatte des Users
 z.B. für Speicherung von Benutzerinfos,
 Passwörtern,
 Umgebungseinstellungen,
 Formulardaten per Cookies
 der Verfolgung der Surfgewohnheiten des Users dienen
 von lokal ablaufenden Webseiten (ohne Internetzugriff) verwendet werden
 müssen nicht Bestandteil einer Webseite sein, da sie kein Bestandteil von HTML sind
 haben vorrangig einen Nutzen, der im Interesse des Webseiten-Anbieters und weniger im Interesse des User liegt.

Cookies können, wenn sie als Voraussetzung für die Lauffähigkeit einer Webseite verwendet werden, für den User eine Nötigung darstellen:

Eine Nötigung liegt nicht vor, solange der User die Wahl hat, die Webseite besuchen zu können bzw. der User in die vollendete Lage versetzt ist, die Cookieverwaltung der Webseite eigenständig und damit die Wirksamkeit von Cookies vorab unterbinden zu können.
 Dagegen ist das automatische Aufpoppen einer Webseite mit Cookieverwaltung pure Nötigung, wenn der User das Aufpoppen bzw. die Cookieverwaltung objektiv nicht verhindern kann.

Die Zielsetzung der Cookie-Verwendung ist im Regelfall durch den User nicht ersichtlich, da Cookies zwar ein einheitliches Format besitzen, aber beliebige Inhalte, die dem User nicht unbedingt vorab zur Cookieverwendung zur Kenntnis gelangen. Als Missbrauch sind Cookies einzuschätzen, die kein Verfallsdatum besitzen und/oder ohne Kenntnis des Users permanent auf der Festplatte existieren.

Die Kenntnisse des Users, weshalb und in welcher Form Cookies verwendet werden, ist Sache des Users, wenn der Hersteller der Browsersoftware bzw. der Hersteller der Webseite keine Aufklärung des Users zu den verwendeten Cookies betreibt. Das Risiko der Cookieverwendung liegt beim User, wenn der User nicht in die vollendete Lage versetzt ist, sich mit für ihn üblichen und seiner Kenntnis zugänglichen Mitteln gegen die Cookie-Verwaltung zu wehren. Die Verhältnismäßigkeit der Cookie-Verwendung zum Surfzweck des Users lässt sich z.T. anzweifeln. Im letzteren Fall ist die Cookieverwendung eine aggressives Verhalten des Webseitenanbieters gegen über dem User und dessen Eigentum, wenn für den User ein Risiko auf Erleiden von Schaden besteht und der User das Risiko sowie den Schaden im Voraus nicht abschätzen kann. Cookieverwendung ohne Transparenz für den User kann aus Usersicht als Risiko und als unerwünscht eingestuft werden
 als Eingriff in die Privatsphäre und in das Datenschutzbedürfnis des Users aufgefasst werden
 rechtlich relevant werden.

Der User sollte wissen, ob und wie der Browser Cookies zulässt. Es besteht die Gefahr, dass der User aufgrund der weit verbreiteten Cookie-Nutzung von Webseitenanbietern die Cookieverwendung generell zulässt, um nervende Meldungen oder eventuelle Surf-Misserfolge bei ohne Cookies nicht lauffähigen Seiten zu vermeiden. Der Einsatz einer Firewall erfordert zusätzliche Massnahmen und Kenntnisse des Users, um den Cookiezugriff steuern zu können, wenn die Firewall-Software nicht Bestandteil des Software ist, die der User beim Surfen nutzt, bzw. die Firewall-Software unverhältnismäßig kompliziert zum Surfzweck des Users ist. Es ist dem User anzuraten, Cookies nach einer Online-Sitzung auf der Festplatte physisch zu löschen bzw. Cookies über Browsereinstellungen und/oder über eine Firewall zu steuern (Cookiefreigabe vorab nur für durch den User bewusst ausgewählte Webseiten).

Ein Analogon zur Bedeutung der Cookie-Nutzung ist die Verlinkung auf fremde Webseiten-Angebote:

Die **pauschale** Erklärung des Webseitenanbieters zur ausdrücklichen Distanzierung von Inhalten zu verlinkten Fremdanbieterseiten
 (und damit auch von der möglichen Cookieverwaltung durch Fremdanbieter) ist inzwischen eine **veraltete** Rechtsnorm geworden und sieht beispielhaft wie folgt aus:

"Für alle Links dieser gesamten Homepage gilt: Ich möchte ausdrücklich betonen, daß ich keinerlei Einfluß auf die Gestaltung und die Inhalte der gelinkten Seiten habe. Deshalb distanzieren ich mich hiermit ausdrücklich von allen Inhalten aller von dieser Homepage aus gelinkten Seiten."

Dabei ist es unerheblich, welcher Fremdanbieter (z.B. eine Partner-Webseite) betroffen ist.

Es **kann** aber der Fall eintreten, dass die pauschale Distanzierung nichts anderes als eine leider übliche Form des Weiterreichens des Risikos an den User und unabhängig davon ist, welche realen Voraussetzungen sich zum Zeitpunkt des Webseitenbesuches durch den User vorfinden lassen würden bzw. vorgefunden werden bzw. wurden. Die Zweizügigkeit und Praxisferne der pauschalen Distanzierung lässt sich salopp auch so sagen: Irren ist menschlich, vorallem für den Endverbraucher in der Kette, den User.

Achtung: Zu prüfen ist auch immer, ob überhaupt eine Verlinkung generell oder im Einzelfall stattfinden darf, da Rechtsnormen **permanent** und nicht unbedingt logisch oder gemäß den üblichen Gepflogenheiten im Web verändert werden: Missbräuchliche Webnutzung ist nicht der Regelfall, kann aber in der Rechtspflege als schwerwiegende, reale oder wahrscheinliche Ausnahme unterstellt werden, die eine juristische Regulierung des **Regelfalls** herbeiführt.

Aufgrund der Uneinheitlichkeit der Rechtspflege ist der Inhaber einer Webseite im unangenehmen Zugzwang: Es bestehen eventuell Interessenskonflikte zwischen dem Anliegen des Webseiteninhabers und den Rechtsnormen (aus Sicht der den Regelfall dominierenden Ausnahme des Regelfalles). Desweiteren ist davon auszugehen, dass private Webseitenanbieter ohne kommerzielle Interessen mit der Verfolgung der Rechtsnormenveränderung überfordert sind und dann das Risiko der Veröffentlichung der Webseite(n) nicht wissen und trotzdem rechtskonform auftreten **müssen**.

Die einfachste Lösung, das Rechtsproblem zu umgehen, ist die Darstellung des Links ohne Klickbarkeit, also ohne seine Aktivierung durch User-Klick auf das Link-Element des Dokumentes in der Webseite (z.B. anstelle der Link-Aktivierung ein



Popup-Fenster anzeigen, das den Link kommentiert, aber nicht aktiviert, also einen Link-Tip offeriert).

Wenn der User die Url selbst aktiviert, z.B. durch Eingabe in der Kommandozeile des Browsers, ist der User verantwortlich. Problem dabei: Konformität des im Link-Tip angegebenen Ziels zu den Rechtsnormen und Kenntnis des User darüber (Ausschluss der Irreführung/Täuschung des Users).

Tip: Die Realisierung eines Gästebuches unterliegt gleichen Kriterien, wenn der Gästebucheintrag durch den Inhaber des Gästebuches unbesehen, weil automatisch, im Gästebuch publiziert wird und der Inhaber des Gästebuches danach Kenntnis zu diesem Eintrag erlangt, der z.B. einen durch den Leser des Gästebuches aktivierbaren Link enthält.

Die Nutzung von Cookies sollte bezüglich Zweck, Dauer und Inhalt dem User mitgeteilt werden und nachvollziehbar sein. Erfahrungsgemäß ist das **nicht** der Regelfall. Auch die Möglichkeit der Browsersoftware, die Cookie-Verwaltung transparent zu halten, entlastet nicht. Der Regelfall ist, dass der User entweder von Cookies keine Kenntnis oder Cookies zum ungestörten Surfen genehmigt hat. Damit tritt nur selten der Fall ein, dass eine vom Webseiten-Inhaber programmierte Meldung zur Aktivierung der abgeschalteten Cookieverwaltung und zum Grund der Cookienutzung auftreten kann. Zumal es für den User eine Zumutung ist, die vom Webseiteninhaber gewollte Cookieverwendung nachvollziehen zu können/müssen, wenn der User die Webseite, die Cookies verwendet, besuchen will. Nochmals sie darauf hingewiesen: Der User trägt das eventuelle Risiko und den Aufwand, die der Webseitenanbieter bewirken könnte.

Hinweis: Als zwingend rechtlich geboten ist die Unterlassung der Darstellung fremder Webseiten in einem Frame, der kein eigenständiges Fenster außerhalb des Kontextes der aufrufenden Webseite darstellt, wenn der Fremdanbieter nicht explizit mit der Frame-Darstellung einverstanden ist. Letzteres ist nicht zu erwarten, da in der Praxis die o.g. pauschale Erklärung oft ins Leere führen wird (weil es schwer nachprüfbar ist, welche Frame-Darstellungen und deren jeweiliger Kontext existieren) und im Moment der Risiko- und Schadenssituation für den User irrelevant, weil nicht helfend ist.

4.3.2.2.4.2.2.3.2. Lage der Cookies am Beispiel von Microsoft Windows 9.x

liegen unter \Windows in einem speziellen Ordner
 meist C:\Windows\Cookies (Ordner, der systemgeschützt ist)
 und/oder im Browser-Cache (Ordner, der systemgeschützt ist)

Hinweis: Der Windows-Pfad mit Laufwerksangabe ist als System-Umgebungs-Variable hinterlegt, auch wenn ein anderes Verzeichnis als \Windows benutzt wird.

Ein Cookie kann bis zu 4 KBytes groß sein.
 Es sind bis zu 300 verschiedene Cookies gleichzeitig speicherbar.

4.3.2.2.4.2.2.3.3. Vermeidung von Cookies

Cookieabschaltung für Onlinesitzung(en)

Wenn die Cookiesverwaltung nicht erwünscht ist, so
 muss sie im Browser abgeschaltet werden, falls es der Browser zulässt
 sollte eine Firewall-Software benutzt werden, wenn diese im Betriebssystem nicht bereits integriert ist bzw. die betriebssystemeigene Firewall bzw. die Browsereinstellungen bezüglich der Cookieverwaltung unzureichend sind.

Achtung: Eine permanente Cookie-Abschaltung bewirkt bei manchen Webseiten, dass diese nicht lauffähig sind, obwohl Cookies kein Bestandteil von HTML sind.

Cookies löschen nach einer Onlinesitzung

nach der Onlinesitzung Löschung des INHALTES
 des Cookie-Ordners: keine DAT-Dateien
 keine system-geschützte Dateien
 keinesfalls des Ordners selbst
 und des Browser-Cache: keine system-geschützte Dateien
 keinesfalls des Ordners selbst

Sichere Löschung kann nur z.T. über Features des Browsers erfolgen (je nach Browserhersteller und Browser-Version).

Falls der Browser keine Features besitzt, so sollte man anstelle der manuellen Löschung vorhandene Features des Betriebssystems bzw. Fremdsoftware verwenden.

4.3.2.2.4.2.2.3.4. Cookie verwalten (Beispiele)

Cookie schreiben

Im Beispiel wird davon ausgegangen, dass NUR der Wert des Cookies und keine Cookiekomponenten geschrieben werden sollen.

```
function schreibe_cookie(freier_bezeichner,wert_des_cookie,anzahl_der_tage)
{
    // Verfalldatum als Datum-Variable erzeugen
    verfall_datum= new Date();

    //Verfalldatum-Variable mit Wert belegen
    verfall_datum.setTime(        verfall_datum.getTime()
                                + (86400000*anzahl_der_tage)
                                );
    // setTime verlangt Millisekunden
    // pro Tag 86400000 Millisekunden = 24 * 60 * 60 * 1000)
    // getTime Zeit des Schreibens
```




```
// wenn anzahl_der_tage=0 ist sofortiges Löschen des Cookie
// nach der Online-Sitzung

// Cookie-Format erzeugen (nur Wert und Verfalldatum)
// keine Blanks in das Format einbauen !!
document.cookie= freier_bezeichner
                + "="
                + escape(wert_des_cookie)
                + ";expires="
                + verfall_datum.toGMTString();
}
```

Cookie lesen

Im Beispiel wird davon ausgegangen: Wenn mindestens eine Komponente vorhanden ist, so wird nur die ERSTE gelesen.

```
function lese_cookie(freier_bezeichner)
{
    // Funktionswert initialisieren
    // Annahme: Cookie NICHT gefunden
    funktionswert=null;

    // Suchbegriff anhand Cookie-Name bilden
    // darf keine Blanks enthalten !!
    suchebegriff=freier_bezeichner + "=";
    // und die Länge des Suchbegriffes holen
    suchbegriff_laenge=suchbegriff.length;

    // Anzahl der Bytes aller Cookies im Puffer des Dokumentes holen
    puffer_laenge=document.cookie.length;

    // und suchen: Pufferinhalt wird häppchenweise durchsucht per Teilkette
    teilkette_startposition=0; // Startposition der Teilkette im Puffer

    while (teilkette_startposition < puffer_laenge) // solange nicht Pufferende erkannt
    {
        // Pufferinhalt als Häppchen, also Teilkette adressieren
        teilkette_endeposition=teilkette_startposition + suchbegriff_laenge;

        // Pufferinhalt als Häppchen holen
        teilkette=document.cookie.substring(teilkette_startposition,teilkette_endeposition);

        // Vergleich der Teilkette mit dem Suchbegriff
        if (teilkette == suchebegriff)
        {
            // Cookie wurde gefunden
            // Rest der Teilkette (Häppchen) nach einem Semikolon durchsuchen
            // also prüfen, ob mindestens eine Komponente vorhanden ist
            semi_position= document.cookie.indexOf(";",teilkette_endeposition);

            if (semi_position == -1)
            {
                // kein Semikolon gefunden wenn NICHT >=0
                funktionswert=
                unescape(document.cookie.substring(
                    teilkette_endeposition,document.cookie.length
                ));
            }
            else
            {
                // Semikolon gefunden
                // NUR die ERSTE Komponente lesen
                funktionswert=
                unescape(document.cookie.substring(
                    teilkette_endeposition,semi_position
                ));
            }
        }

        return(funktionswert); // gesamte Funktion beenden
    }
    else
    {

```



```

        // Cookie wurde NICHT gefunden
        // Trennblank zum nächsten Cookie suchen
        blank_position=
            document.cookie.indexOf(" ",teilkette_startposition);
        if (blank_position == -1)
        {break;} // kein weiteres Cookie vorhanden, also while abbrechen
        else
        {teilkette_startposition= blank_position+1;}
        // neue Startposition der Teilkette holen
    }
}
return(funktionswert);
}

```

4.3.2.2.4.2.3.5. document.cookie Collection

Feld der Zeiger auf alle Cookies zum Dokument

Ein Cookie kann bis zu 4 KBytes groß sein.

Es sind bis zu 300 verschiedene Cookies gleichzeitig speicherbar.

Cookies haben den gleichen Aufbau aber verschiedene Inhalte:

Optionale Eigenschaften müssen ein Semikolon vorgesetzt bekommen und können beliebig kombiniert werden.

Erzeugung:

durch den Browser

Zugriff:

```

[ var ZeigerAufFeld = ] document.cookie
[ var ZeigerAufFeldElement = ] document.cookie[Index]

```

Index	Integer ab 0 muss in [] kodiert sein
-------	--

Ministdeklaration eines Cookie (Wert eines Cookies):

```

document.cookie[0] = wert_des_cookie_1;

oder

var cookie_feld = document.cookie;
cookie_feld[0] = wert_des_cookie_1;

wert_des_cookies:  Zeichenkette
                    kann HTML-gerecht kodierte Sonderzeichen enthalten
                    darf kein Blank enthalten, da jedes Cookie intern als Endekennzeichen ein Blank
                    besitzt
Aufbau:
                    freierwert[;eigenschaften_liste]

                    Semikolon trennt die optionale Eigenschaftenliste ab
                    Eigenschaften in der Liste durch Semikolon trennen

                    freierwert: cookie_titel=freier_wert

                    = ist festkodiert

                    Bsp: "Filmbeginn=23:15;path=...;expires=....."

```

Anwendungsbereich des Cookie also nicht die Lage auf der Festplatte der Users:

```

path=pfad_angabe
path= ist festkodiert
pfad_angabe:
    Bsp.: 'Otto=der_erste;c:\privat\html_files\'
           also Anwendbarkeit nur möglich, wenn HTML-Dokument
           innerhalb bzw. unterhalb von c:\privat\html_files liegt
           wenn nicht kodiert, wo wird der Pfad des Browser-Cache verwendet

```

Servergruppezugehörigkeit:

```

domain=server_name_fragment
domain= ist festkodiert
server_name_fragment:
    Bsp.: "Otto=der_erste;domain=online.de"
           also alle Server, die im Namen den Rest
           online.de haben
           wenn nicht kodiert, so wird der volle Name
           von dem Server abgelegt, der das Cookie

```



verwaltet und nutzt

Verfallsdatum:**expires=datum_angabe**

expires= ist festkodiert

datum_angabe: wochentag, tt-mm-jj hh-mm-ss
nach Greenwich-Time

Bsp.: var loeschdatum = new Date();

"Otto=der_erste;epires=" +

loeschdatum.toGMTString();

wenn nicht kodiert, so Cookie mit Ende der

Onlinesitzung von der Festplatte des

Users automatisch gelöscht

wenn kodiert, so am Verfallsdatum bzw. danach

von der Festplatte des Users automatisch

gelöscht (danach, wenn der User des PC

nach dem Verfallsdatum erst wieder

nutzt)

Cookie nur senden, wenn HTTPS-Protokoll also sichere Verbindung benutzt wird:**secure**

kein Wert

Beispiel für Cookie verwalten:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookien_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {

```



```

        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

return CookieWert;
}
</SCRIPT>

```

Eigenschaften:

.length Anzahl der Cookies, ab 1

Methoden:

keine

Auf den Wert des Cookies sind alle Methoden des String-Objektes anwendbar, da der Wert immer eine Zeichenkette ist.

4.3.2.2.4.2.2.4. **Zusätzliche Verhaltensweisen eines HTML-Elementes bzw. des HTML-Dokumentes** im Internet Explorer ab 5.x

Nachfolgend werden Möglichkeiten der Implementation von zusätzlichen Verhaltensweisen vom HTML-Dokument und/oder HTML-Elemente im HTML-Bokument beschrieben, die nur der Internet Explorer ab 5.x aufweist.

Behaviors sind Verhaltensweisen von Elementen.

Verhaltensweisen werden mit externen *.htc-Dateien beschrieben, die importiert werden müssen.

Nur die Standard-Behavior (z.B. .style.time2) des IE benötigen keine *.htc-Datei

4.3.2.2.4.2.2.4.1. *Standard-Behavior*

im Browser implementiert Standard-Verhaltensweisen z.B. anhand des Objektes style.time2 (siehe dort)

4.3.2.2.4.2.2.4.2. *element Objekt des Internet Explorer (HTC-Datei)*

Objekt einer HTC-Komponente aus einer HTC-Datei

ab IE 5.x

Behavior einbinden:

Beispiel 1:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne () // Bezug über ID_Event
    {
        var EventObjekt    = createEventObject();
        EventObjekt.result = "Wert";

        ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>

    @media all { privater_tag_namensraum\;privater_tag {behavior:url(test.htc)} }

</STYLE>
</HEAD>
<BODY>
<privater_tag_namensraum:privater_tag ID="ID_privater_tag"
    onResultChange="ID_Div.innerText=window.event.result"
>
    <TABLE>
        <TR>
            <DIV
                ID="ID_Div"
                ALIGN=RIGHT
                STYLE="border: '.025cm solid gray'"
            >
                0.
            </DIV>
        </TR>
    </TABLE>

```



```

<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" % " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" + ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" = ">
  </TD>
</TR>

```

</TABLE>

</privater_tag_namensraum:privater_tag>

<BODY>



</HTML>

Beispiel 2:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:METHOD NAME="private function" />
    <SCRIPT LANGUAGE="JScript" >
      function private function ()
      {
        // .....
      }
    </SCRIPT>
  </PUBLIC:METHOD>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML>
<HEAD>
  <STYLE>
    .klasse {behavior:url(test.htc)}
  </STYLE>
</HEAD>
<BODY ID="ID_Body"
  CLASS="klasse"
>
  <DIV onclick=" ID_Body.private function ()">
    .....
  </DIV>
</BODY>
</HTML>

```

HTC-Datei:

dient der Implementierung von Verhaltensweisen (Behavior) in das HTML-Dokument

Suffix *.htc

beinhaltet Script und HTC-Komponenten

HTC-Komponente:

dient der Implementierung von Verhaltensweisen (Behavior) in das HTML-Dokument

ist Bestandteil der HTC-Datei

HTC-Komponenten als element-Objekt

erzeugen per PUBLIC:COMPONENT (siehe dort)

HTC-Komponente als Objekt in der Collection document.all im HTML-Dokument

erzeugen per PUBLIC:PROPERTY (siehe dort)

HTC-Komponente	kann	besitzen:	Eigenschaften	(PUBLIC:PROPERTY)
			Methoden	(PUBLIC:METHOD)
			Event	(PUBLIC:EVENT)
			Eventhandler	(PUBLIC:ATTACH)
			JScript-Code	
	verarbeiten		Event	(PUBLIC:ATTACH und PUBLIC:EVENT)

ist der Container (PUBLIC:COMPONENT) der o.g. Bestandteile für die Kodierung in der HTC-Datei

PUBLIC:ATTACH

für HTC-Komponente laut PUBLIC:COMPONENT einem Event den Eventhandler zuordnen und diese Zuordnung dem HTML-Dokument bekanntgeben

Mehrfachkodierung für ein und dasselbe Event vermeiden, da sonst Zufallsauswahl erfolgt
Syntax:

```

<PUBLIC:ATTACH EVENT = Kette1
  [FOR = Kette2]
  [ID = Kette3]
  ONEVENT = Kette4
>
</PUBLIC:ATTACH>

```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

Kette1 Bezeichner eines Events in der Form onXXX
laut HTC-Referenz
Achtung: Gross-Klein wird unterschieden



Kette2 optional
Referenz auf die Eventquelle
Standard: Objekt, das das Behavior besitzt
auch kodierbar "document" oder "window" kodierbar

Kette3 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette4 Bezeichner der Eventhandler-Script-Funktion
in der Funktion die Referenzen auf Methoden der Objekte wie window
oder document immer im Pfad komplett kodieren also
mit window. bzw. document.

Beispiel:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT="EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor      = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu);           // nicht () kodieren !!!
        detachEvent ('onmouseout', CursorNormal);          // nicht () kodieren !!
    }

    attachEvent ('onmouseover', CursorNeu);
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

PUBLIC:COMPONENT HTC-Komponente definieren (Container) und dem HTML-Dokument bekanntgeben
Kinder: einmalig **PUBLIC:DEFAULTS**
ein-oder mehrmalig **PUBLIC:ATTACH**,
PUBLIC:EVENT,
PUBLIC:METHOD,
PUBLIC:PROPERTY

Syntax:

```
<PUBLIC:COMPONENT
    [ID = Kette1]
    [lightWeight      = Kette2]
    [literalContent   = Kette3]
    [NAME             = Kette4]
    [supportsEditMode = Kette5]
    [tagName          = Kette6]
    [URN              = Kette7]
>
    <!-- hier alle anderen HTC-Komponenten kodieren -->
</PUBLIC:COMPONENT>
```

Kette1 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette2 optional
"true" kein Parsen und eventuelle Anzeige aller Tags



(Inhalte zwischen den Start- und Endetags)
 "false" Standard
 Parsen und eventuelle Anzeige aller Tags
 (Inhalte zwischen den Start- und Endetags)
 Parsen und Anzeige ein/aus für einen speziellen Tag: siehe Kette3

- Kette3 optional
 Auswertung des Inhaltes zwischenTags laut tagName=Kette6
 "false" Standard
 Inhalt zwischen Start- und Ende-Tag laut Kette6 wird geparkt und eventuell angezeigt
 "nested" ab IE 6.x
 Inhalt zwischen dem **ersten** Start- und dem **letzten** Ende-Tag laut Kette6 wird nicht geparkt und nicht angezeigt, sondern als Dateninsel (Datenquelle) verarbeitet
 "true" Inhalt zwischen dem **ersten** Start- und dem **ersten** Ende-Tag laut Kette6 wird nicht geparkt und nicht angezeigt, sondern als Dateninsel (Datenquelle) verarbeitet
- Kette4 optional
 analog zu ID
- Kette5 optional
 generelle Veränderbarkeit des Kontextes aller HTC-Komponenten
 "true" veränderbar
 "false" Standard
 nicht veränderbar
- Kette6 optional
 Bezeichner eines Tags einer oder mehrerer HTC-Komponenten
 Verwendung des Tags : siehe literalContent = Kette3
 userdefinierter Tag ab IE 5.5
- Kette7 optional
 Uniform Resource Name (URN) für Eventsteuerung:
 Quelle des Events (srcUrn) laut URN des Behavior,
 der das Event erzeugt

PUBLIC:DEFAULTS

für eine HTC-Komponenten die Standardangaben setzen und dem HTML-Dokument bekanntgeben
 darf nur 1x kodiert werden im Container PUBLIC:COMPONENT
 Syntax:

```
<PUBLIC:DEFAULTS
  [canHaveHTML      = Kette1]
  [contentEditable  = Kette2]
  [style             = Kette3]
  [tabStop           = Kette4]
  [viewInheritStyle = Kette5]
  [viewLinkContent  = Kette6]
  [viewMasterTab    = Kette7]
>
</PUBLIC:DEFAULTS>
```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

- Kette1 optional
 "false" Komponente darf zwischen Start- und Endetag kein HTML besitzen
 "true" Komponente darf zwischen Start- und Endetag HTML besitzen
- Kette2 optional
 "inherit" Veränderlichkeit des Kontext von Eltern geerbt
 "false" Veränderlichkeit nicht möglich
 "true" Veränderlichkeit möglich
- Kette3 optional
 Werte wie laut HTML-STYLE-Attribut
- Kette4 optional



"false"	Standard
"true"	Komponente per TAB-Taste nicht aktivierbar
"true"	Komponente per TAB-Taste aktivierbar
Kette5	optional
"false"	Styles vom HTML-Dokument nicht übernehmen
"true"	Styles vom HTML-Dokument übernehmen
Kette6	optional
"false"	Standard
"true"	Viewlink nicht benutzt
"true"	Viewlink nutzbar
Kette7	optional
"false"	Master-Element des Viewlink in der Tab-Tastenfolge des HTML-Dokumentes nicht enthalten
"true"	Master-Element des Viewlink in der Tab-Tastenfolge des HTML-Dokumentes enthalten

PUBLIC:EVENT für eine HTC-Komponente ein Event definieren und dem HTML-Dokument bekanntgeben
Syntax:

```
<PUBLIC:EVENT
  [ID      = Kette1]
  NAME     = Kette2
>
</PUBLIC:EVENT>
```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

Kette1 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette2 analog zu ID

Methode: Event erzeugen und an das HTML-Dokument weiterreichen

Syntax:

```
Kette1.fire( [zeiger_auf_event_objekt] )
```

zeiger_auf_event_objekt per createEventObject() erzeugen

Beispiel:

test.htc enthält:

```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne()// Bezug über ID_Event
    {
      var EventObjekt = createEventObject();
      EventObjekt.result = "Wert";

      ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>
```

HTML-Dokument enthält:

```
<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
  @media all { privater_tag_namensraum:privater_tag { behavior:url(test.htc) } }
</STYLE>
</HEAD>
<BODY>
  <privater_tag_namensraum:privater_tag ID="ID_privater_tag"
    onResultChange="ID_Div.innerText=window.event.result"
  >
  <TABLE>
```



```

<TR>
  <DIV ID="ID_Div"
    ALIGN=RIGHT
    STYLE="border: '.025cm solid gray'"
  >
    0.
  </DIV>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" % " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" + ">

```




```

</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

PUBLIC:METHOD für ein HTC-Komponente eine Methode definieren und dem HTML-Dokument bekanntgeben
Syntax:

```

<PUBLIC:METHOD
  [ID = Kette1]
  [INTERNALNAME = Kette2]
  NAME = "sName"
>
<!-- Script-Code der Methode mit Bezeichner laut Kette3 //-->
</PUBLIC:METHOD>

```

muss im Container PUBLIC:COMPONENT kodiert sein

Kette1	optional kein Standard Funktionalität wie ID-Attribut im HTML-Tag
Kette2	optional komponenten-lokaler Bezeichner der Methode Standard ist Kette3
Kette3	öffentlicher Name der Methode im HTML-Dokument analog zu ID

Beispiel:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:METHOD NAME="private function" />
  <SCRIPT LANGUAGE="JScript" >
    function private function ()
    {
      // .....
    }
  </SCRIPT>
</PUBLIC:METHOD>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML>
<HEAD>
  <STYLE>
    .klasse {behavior:url(test.htc)}
  </STYLE>
</HEAD>
<BODY ID="ID_Body"
  CLASS="klasse"
>
  <DIV onclick="ID_Body.private function ()">
    .....
  </DIV>
</BODY>
</HTML>

```

PUBLIC:PROPERTY für eine HTC-Komponente eine Eigenschaften definieren
und dem HTML-Dokument bekanntgeben
und automatisch als Element der Collection document.all erzeugen
Zugriff auf HTC-Komponente im HTML-Dokument:

```

element.document.all(name_der_komponente)
                        name_der_komponente      laut Attribut NAME

```

Syntax:

```

<PUBLIC:PROPERTY
  [GET = Kette1]

```



```

[ID] = Kette2]
[INTERNALNAME = Kette3]
[NAME = Kette4]
[PERSIST = Kette5]
[PUT = Kette6]
[VALUE = Kette7]
>
</PUBLIC:PROPERTY>

```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

Kette1 optional
Bezeichner der Funktion, die im HTML-Dokument aufzurufen ist, wenn der Wert der HTC-Komponenten-Eigenschaft ermittelt werden soll (Lesen der HTC-Komponenten-Eigenschaft)

Kette2 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette3 optional
komponenten-lokaler Bezeichner der Methode
Standard ist Kette4

Kette4 öffentlicher Name der Komponente im HTML-Dokument
analog zu ID

Kette5 optional
"true" Eigenschaft dem HTML-Dokument bekanntgeben
"false" Eigenschaft dem HTML-Dokument nicht bekanntgeben

Kette6 optional
Bezeichner der Funktion, die im HTML-Dokument aufzurufen ist, wenn der Wert der HTC-Komponenten-Eigenschaft gesetzt werden soll (Schreiben der HTC-Komponenten-Eigenschaft)

Kette7 optional
Initial-Wert bzw. aktueller Wert der Eigenschaft

Methode: Kette2.fireChange() erzeugt Event onpropertychange und gibt es an das HTML-Dokument, sobald der Wert der Eigenschaft verändert wird, also PUT aktiviert wird
muss in der Routine zu PUT kodiert werden

Beispiel 1:

```

<PUBLIC:PROPERTY NAME="HTC_Komponente"/>
<PUBLIC:ATTACH EVENT="onclick" ONEVENT="EventHandler()" />

<SCRIPT LANGUAGE="JScript">
function EventHandler()
{
    var HTC_KomponenteObjekt = element.document.all(HTC_Komponente);

    var HTC_KomponenteObjekt _Sichtbarkeit_RunTimeStyle =
        HTC_KomponenteObjekt.runtimeStyle.display;

    if (HTC_KomponenteObjekt _Sichtbarkeit_RunTimeStyle == "none")
    {
        runtimeStyle.listStyleImage = "url('test1.gif')";
        HTC_KomponenteObjekt _Sichtbarkeit_RunTimeStyle = "";
    }
    else
    {
        runtimeStyle.listStyleImage = "url('test2.gif')";
        HTC_KomponenteObjekt _Sichtbarkeit_RunTimeStyle = "none";
    }
}
</SCRIPT>

```



Beispiel 2:

```
<PUBLIC:PROPERTY
    ID="ID_HTC_Komponente"
    NAME="HTC_Komponente"
    PUT="Eigenschaft_Schreiben"
    GET="Eigenschaft_Lesen"
/>

<SCRIPT LANGUAGE="JScript">
    var Eigenschaft_Wert =null;

    function Eigenschaft_Schreiben (Wert)
    {
        // Wert zuweisen
        Eigenschaft_Wert = Wert;

        // und Event erzeugen
        ID_HTC_Komponente.fireChange();
    }

    function Eigenschaft_Lesen ()
    {return Eigenschaft_Wert; }
</SCRIPT>
```

Methoden:

.createEventObject() Event erzeugen in einem Eventhandler einer HTC-Komponente
 siehe Objekt element und HTC-Datei
 Syntax:

```
[ var Zeiger = ] createEventObject()
```

Zeiger auf Eventobjekt

Beispiel:

test.htc enthält:

```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
    <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

    <SCRIPT LANGUAGE="JScript">
        function Berechne()// Bezug über ID_Event
        {
            var EventObjekt      = createEventObject();
            EventObjekt.result      = "Wert";

            ID_Event.fire (EventObjekt);
        }
    </SCRIPT>
</PUBLIC:COMPONENT>
```

HTML-Dokument enthält:

```
<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>

    @media all { privater_tag_namensraum\;privater_tag {behavior:url(test.htc)} }

</STYLE>
</HEAD>
<BODY>
<privater_tag_namensraum:privater_tag      ID="ID_privater_tag"
                                             onResultChange="ID_Div.innerText=window.event.result"
>

    <TABLE>
        <TR>
            <DIV      ID="ID_Div"
                ALIGN=RIGHT
                STYLE="border: '.025cm solid gray'"
            >
                0.
            </DIV>
        </TR>
        <TR>
            <TD>
```



```
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
</TD>
</TR>
<TR>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" % " DISABLED>
</TD>
</TR>
<TR>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
</TD>
</TR>
<TR>
<TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" + ">
</TD>
<TD>
    <INPUT TYPE=BUTTON VALUE=" = ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>
```



.fire() ein in der HTC-Komponente definiertes Event erzeugen und an das HTML-Dokument weiterreichen
 siehe Objekt element und HTC-Datei
 Syntax:

ID_Event.fire([zeiger_auf_event_objekt])

ID_Event laut Wert des Attributes ID in <PUBLIC:EVENT ...>

zeiger_auf_event_objekt per createEventObject() erzeugen

Beispiel:

test.htc enthält:

```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne()// Bezug über ID_Event
    {
      var EventObjekt = createEventObject();
      EventObjekt.result = "Wert";

      ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>
```

HTML-Dokument enthält:

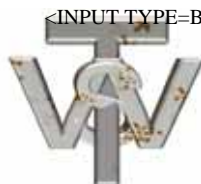
```
<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>

  @media all { privater_tag_namensraum:\:privater_tag {behavior:url(test.htc)} }

</STYLE>
</HEAD>
<BODY>
<privater_tag_namensraum:privater_tag ID="ID_privater_tag"
  onResultChange="ID_Div.innerText=window.event.result"
>

  <TABLE>
    <TR>
      <DIV ID="ID_Div"
        ALIGN=RIGHT
        STYLE="border: '.025cm solid gray'"
      >
        0.
      </DIV>
    </TR>
    <TR>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 7 ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 8 ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 9 ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" / ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" C ">
      </TD>
    </TR>
    <TR>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 4 ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 5 ">
      </TD>
      <TD>
        <INPUT TYPE=BUTTON VALUE=" 6 ">

```



```

</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" * ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" % " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 2 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 3 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" - ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 0 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" +/- ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" . ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" + ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" = ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

`.fireChange()` erzeugt Event `onpropertychange` und gibt es an das HTML-Dokument, sobald der Wert der Eigenschaft per Aufruf der Funktion laut Attribut `PUT` verändert wird
 Aufruf von `.fireChange()` muss programmiert werden
 siehe Objekt `element` und HTC-Datei
 Syntax:

```
ID_HTC_Komponente.fireChange();
```

`ID_HTC_Komponente` laut Attribut `ID` in `<PUBLIC:PROPERTY ...>`

Beispiel:

```

<PUBLIC:PROPERTY
  ID="ID_HTC_Komponente"
  NAME="HTC_Komponente"
  PUT="Eigenschaft_Schreiben"
  GET="Eigenschaft_Lesen"
/>

<SCRIPT LANGUAGE="JScript">
  var Eigenschaft_Wert =null;

  function Eigenschaft_Schreiben (Wert)
  {
    // Wert zuweisen
    Eigenschaft_Wert = Wert;
  }

```



```

        // und Event erzeugen
        ID_HTC_Komponente.fireChange();
    }

    function Eigenschaft_Lesen ()
    {return Eigenschaft_Wert; }
</SCRIPT>

```

Events:

für <PUBLIC:ATTACH EVENT => (außer onpropertychange)

oncontentready erzeugt, wenn auf die HTC-Komponente zugegriffen wird und dieses komplett geparkt ist und das Event window.onload erzeugt wurde
Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden.
siehe Objekt element und HTC-Datei

Beispiel:

```

<PUBLIC:ATTACH EVENT="oncontentready" ONEVENT="Anzeige()" />
<SCRIPT LANGUAGE="JScript">
    function Anzeige()
    {window.alert ('Objekt element mit innerHTML = ' + element.innerHTML); }
</SCRIPT>

```

oncontentsave erzeugt, wenn die Umgebung der HTC-Komponente gespeichert oder kopiert wird
Hinweis: Änderung einer Eigenschaft der HTC-Komponente wird per Event onpropertychange angezeigt
siehe Objekt element und HTC-Datei

ondetach erzeugt, wenn die Event-Überwachung in einer HTC-Komponente abgeschaltet wird
siehe Objekt element und HTC-Datei

Beispiel:

```

<PUBLIC:ATTACH EVENT="ondetach" ONEVENT="EventEntfernen()"/>
<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor      = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
        detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
    }

    attachEvent ('onmouseover', CursorNeu);
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```

ondocumentready erzeugt, wenn das HTML-Dokument, das die HTC-Komponente implementiert hat, komplett geparkt ist und das Event window.onload erzeugt wurde
Parsen des HTML-Dokumentes schliesst Implementierung der HTC-Datei und Parsen der HTC-Komponenten ein
Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden.
siehe Objekt element und HTC-Datei

onpropertychange erzeugt, wenn die Methode **ID_HTC_Komponente.fireChange()** in der Funktion laut Attribut PUT aufgerufen wird (Aufruf muss programmiert werden), wobei die Funktion eine Eigenschaft der HTC-Komponente mit neuem Wert belegt



siehe Objekt element und HTC-Datei

4.3.2.2.4.2.2.4.3. *behaviorUrns Collection des Internet Explorer*

Feld aller Uniform Resource Name (URN) als String

Behaviors sind Verhaltensweisen von Elementen.

siehe auch document.namespace Objekt

Syntax:

```
[ var ZeigerAufFeld = ] object.behaviorUrns
[ var ZeigerAufElementAusFeld = ] object.behaviorUrns[Index]
```

Index Integer, ab 0
muss in [] kodiert sein

Beispiel

```
<HEAD>
<STYLE>
    DIV { behavior:url(fly.htc) url (zoom.htc) url (fade.htc)}
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        var ZeigerAufFeld = ID_Div.behaviorUrns;
        if (ZeigerAufFeld != null)
        {
            for (var i=0; i < ZeigerAufFeld.length; i++)
            {alert (ZeigerAufFeld.item[i]);}
        }
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeige()">
    <DIV ID="ID_Div">Test </DIV>
</BODY>
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

4.3.2.2.4.2.2.4.4. *document.namespace Objekt des Internet Explorer*

Dieses Objekt dient der dynamischen Verwaltung von Behavior (Verhalten) eines Elementes ab IE 5.5 (Behavior sind ab IE 5.x möglich).

Der Programmierer kann private Tags erzeugen und deren Verhalten (Behavior) selbst bestimmen. Problem dabei ist, dass der Browser private Tags nicht kennt. Daher muss dazu folgendes realisiert werden:

Ein Tag hat im Browser einen Namensraum. Tags die der Browser laut HTML kennt, liegen also im HTML-Namensraum. Private Tags, die natürlich nicht aus dem HTML-Namensraum stammen, müssen also einen eigenen Namensraum erhalten, damit der Browser diese Tags parsen kann.

Ein Tag hat im Browser eine feste Definition. Tags, die aus dem HTML-Namensraum kommen (z.B. HEAD) müssen dem Browser nicht weiter mitgeteilt werden. Er kennt sie und kann sie parsen. Private Tags dagegen müssen als Element des privaten Namensraumes definiert werden.

Ein Tag hat im Browser eine genau definierte Aktion, die nach dem Parsen durch den Browser abgearbeitet wird, wobei der Browser dabei die Abarbeitungsfolge optimiert (siehe DOM). Ein privates Tag muss natürlich auch eine Aktion auslösen, die aber dem Browser erst mitgeteilt werden muss, damit er sie starten kann.

Ein Tag hat im Browser eine genau definierte Position im DOM. DOM ist browserabhängig und stellt die Struktur aller Objekte eines Dokumentes derart dar, dass der Browser objektorientiert das Dokument abarbeiten kann. Ein privates Tag muss natürlich auch Teil vom DOM sein, also integriert werden. Der Ort der Integration ist die Einbettung des Tags in das body Objekt.

Um diese Aktionen in das Dokument einzubetten, werden benutzt

HEAD,
BODY,
Script,
XML

für die Auslagerung der Aktionen des Tags als Programmcode das *.htc-Dateiformat.

Das Feld aller Behavior im Dokument ist die Collection document.namespaces (siehe dort).

Das einzigste Event ist onreadystatechange.



Der Internet Explorer kann von Hause aus Behavior nachladen, die von Microsoft geschrieben wurden. Wird ein Behavior verlangt, so erfolgt der Download der *.htc-Datei.

Nachfolgendes Beispiel zeigt den Import per Script:

```
<HEAD>
<HTML XMLNS: Mein_NamensRaum>
</HEAD>
<BODY onload=StartImport(>
<SCRIPT>
    var NamensRaumObjekt;

    function StartImport()
    {
        // HTML XMLNS füllt die Collection namespaces
        //      da nur ein Namensraum erzeugt wurde, ist also nur 1 Element
        //      in der Collection namespaces

        // Namensraum laut Collection referenzieren
        NamensRaumObjekt = document.namespaces[0];

        // Import des Programmcodes aus der Datei test.htc starten (Download)
        NamensRaumObjekt.doImport("test.htc");

        // prüfen ob Import (Download) beendet ist
        if (NamensRaumObjekt.readyState != "complete")
        {
            // Import läuft noch also Endeereignis abfangen per Eventhandler
            //      wenn Import beendet ist, soll die Einbindung des
            //      Tags in das body-Objekt erfolgen
            NamensRaumObjekt.attachEvent("onreadystatechange",
                TagInDasBodyObjektEinbinden
            ); // Funktion ohne () kodieren !!!
        }
        else
        {
            // Import erledigt, also den Tag in das BODY-Objekt einbinden
            addTagNamesToBody();
        }

        return true;
    }

    function TagInDasBodyObjektEinbinden()
    {
        // prüfen ob Import (Download) beendet ist
        if (NamensRaumObjekt.readyState != "complete")
        {return;}
        else
        {
            // Eventabfangen abschalten
            NamensRaumObjekt.detachEvent( "onreadystatechange",
                TagInDasBodyObjektEinbinden
            );

            // Tagobjekt erzeugen
            //      Tag ist ein Textelement
            var TagObjekt = document.createElement("Mein_NamensRaum:MeinTag");

            // Textelement füllen zum Rendern
            TagObjekt.innerText = "ElementBehavior";

            // und in das body-Objekt einbinden
            document.body.appendChild(TagObjekt);
        }
    }
</SCRIPT>
</BODY>
</HTML>
```

Inhalt der test.htc



```

<public:component tagName=MeinTag>
<public:attach event=onreadystatechange onevent=TagAktion()/>
</public:component>
<script>
    function TagAktion()
    {
        element.document.bgColor = "red";
    }
</script>

```

Nachfolgendes Beispiel für den Import per HTML:

```

<HTML XMLNS:MeinNamensRaum1
XMLNS:MeinNamensRaum2
XMLNS:MeinNamensRaum3
>
<HEAD>
    <?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace1.htc">
    <?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace2.htc">
    <?IMPORT NAMESPACE="MeinNamensRaum3" IMPLEMENTATION="namespace3.htc">

    <SCRIPT>
        function TabelleDynamischErweitern()
        {
            var TabellenZeile;
            var TabellenZelleDerZeile;

            for(var Index = 0; Index < document.namespaces.length; Index ++)
            {
                TabellenZeile = ID_Table.insertRow();

                // Zelle 1 mit Index
                TabellenZelleDerZeile = TabellenZeile.insertCell();
                TabellenZelleDerZeile.align = 'center';
                TabellenZelleDerZeile.innerText = Index; // oder .toString()

                // Zelle 2 mit Item-Methode
                document.namespaces.item(iIndex).name
                TabellenZelleDerZeile = TabellenZeile.insertCell();
                TabellenZelleDerZeile.align = 'center';
                TabellenZelleDerZeile.innerText =
                    document.namespaces.item[Index].name;

                // Zelle 3 mit Index
                TabellenZelleDerZeile = TabellenZeile.insertCell();
                TabellenZelleDerZeile.align = 'center';
                TabellenZelleDerZeile.innerText = document.namespaces[Index].name;
            }
        }
    </SCRIPT>
</HEAD>
<BODY>
    <TABLE ID="ID_Table" BORDER=1>
        <TR>
            <TH>Zelle 1 Index</TH>
            <TH>Zelle 2 Feldelement per Item-Methode</TH>
            <TH>Zelle 3 Feldelement per Index</TH>
        </TR>
    </TABLE>
    <BR>
    <BUTTON onclick="TabelleDynamischErweitern();">Behavior anzeigen</BUTTON>
</BODY>
</HTML>

```

Zugriff:

nur über die Collection document.namespaces

```

var ZeigerAufFeldElement = document.namespaces [Index]
ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

```

Index

Integer ab 0



oder String Name oder ID des Elementes
(analog zu NAME und ID-Attribut)
muss in [] kodiert sein

Eigenschaften:

.name Name des Objektes (nicht ID !!!)
muss beim Formular für alle zu sendenden Felder kodiert sein !!
Element darf nicht per Methode .createElement() erzeugt worden sein
.readyState aktueller Status des Objektes beim Füllen des Objektes mit Daten
.urn Uniform Resource Name (URN) des Dokumentes

Methoden:

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
Hinweis: Abschalten mit Methode .detachEvent()
Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
NICHT verkettet sondern in **Zufallsfolge**, es sei denn
die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.detachEvent() Abschalten des Registrieren eines Events durch Eventhandler
wobei Registrierung mit Methode .attachEvent() aktiviert wurde
Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner
zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt
(also Standardbehandlung aktiv)
.doImport() Start des Import des Programmcodes aus einer *.htc-Datei eines Behavior (Verhaltens)
eines Elementes / Objektes

4.3.2.2.4.2.2.4.5. document.namespaces Collection des Internet Explorer

Feld der Zeiger aller namespace Objekte

Syntax:

```
[ var ZeigerAufFeld = ] document.namespaces
[ var ZeigerAufFeldElement = ] document.namespaces [Index]
```

Index Integer ab 0
oder String Name oder ID des Elementes
(analog zu NAME und ID-Attribut)
muss in [] kodiert sein
ZeigerAufFeldElement
ist null, wenn Feldelement nicht vorhanden

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen
hinzufügen erst nach dem kompletten Laden des Dokumentes
Element erzeugen per Methode .createElement()
.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel:

```
<HTML XMLNS:MeinNamensRaum1
XMLNS:MeinNamensRaum2
XMLNS:MeinNamensRaum3
>
<HEAD>
  <?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace1.htc">
  <?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace2.htc">
  <?IMPORT NAMESPACE="MeinNamensRaum3" IMPLEMENTATION="namespace3.htc">

  <SCRIPT>
    function TabelleDynamischErweitern()
    {
      var TabellenZeile;
      var TabellenZelleDerZeile;

      for(var Index = 0; Index < document.namespaces.length; Index ++)
      {
        TabellenZeile = ID_Table.insertRow();

        // Zelle 1 mit Index
        TabellenZelleDerZeile = TabellenZeile.insertCell();
        TabellenZelleDerZeile.align = 'center';
        TabellenZelleDerZeile.innerText = Index; // oder .toString()

        // Zelle 2 mit Item-Methode
```



```

document.namespaces.item(iIndex).name
TabellenZelleDerZeile = TabellenZeile.insertCell();
TabellenZelleDerZeile.align = 'center';
TabellenZelleDerZeile.innerText = document.namespaces.item[Index].name;

// Zelle 3 mit Index
TabellenZelleDerZeile = TabellenZeile.insertCell();
TabellenZelleDerZeile.align = 'center';
TabellenZelleDerZeile.innerText = document.namespaces[Index].name;
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <TABLE ID="ID_Table" BORDER=1>
        <TR>
            <TH>Zelle 1 Index</TH>
            <TH>Zelle 2 Feldelement per Item-Methode</TH>
            <TH>Zelle 3 Feldelement per Index</TH>
        </TR>
    </TABLE>
    <BR>
    <BUTTON onclick="TabelleDynamischErweitern();" >Behavior anzeigen</BUTTON>
</BODY>
</HTML>

```

4.3.2.2.4.2.2.5. Filter eines HTML-Elementes im Internet Explorer (filter Objekt des Internet Explorer)

4.3.2.2.4.2.2.5.1. Einführung

Der Begriff Filter ist einem typischen Anwendungsbeispiel entliehen: Bekanntlicherweise sind Bildeffekte durch Herausfilterung von Bildkomponenten wie Farbe etc. sehr beliebt, so wie es auch ein Bildverarbeitungsprogramm mit Filter-Plugins anbietet. Als dynamisches Ergebnis der Bildverarbeitung ist z.B. die GIF-Datei (*.gif) zu nennen, deren Animationsumfang aber begrenzt ist.

Der Internet Explorer unterstützt dynamische Filter zur Laufzeit des HTML-Dokumentes, wobei nicht **nur** Bild-Objekte animiert werden können. Selbst auf den Einsatz von GIF-Dateien kann verzichtet werden.

Filter sind ebenfalls zusätzliche Verhaltensweisen von HTML-Elementen.

Filter sind nicht per externer Datei definierbar, sondern komplett im Browser implementiert.

Der Internet Explorer bietet Filter in verschiedenen Formen an:

- Objekt filter Es muss vom Objekt, das animiert werden soll, unterstützt werden.
Ein Objekt hat einen genau definierten Umfang an unterstützten Filtern, z.T. gar keine Filterunterstützung
- Objekt .style.time2.transitionFilter Behavior-Objekt des Standard-Beavior .style.time2
Dieser Filter unterstützt diverse HTML-Elemente, auch diejenigen, die das filter Objekt nicht unterstützen.

Nachfolgend wird das **Objekt filter** beschrieben (style.time2.transitionFilter siehe dort)

Das Objekt filter ist ein symbolisches Objekt. Es umfasst diverse Filterarten, deren wichtigsten nachfolgend beschrieben werden.

Filter dienen zur sichtbaren Animation eines Objektes, das selbst **sichtbar sein muss**.

Nachteile von Filtern:

Filter benutzen z.T. intensiv CPU-Ressourcen, z.B. der Fade-Filter (Blenden von Bildern). Da dem Programmierer nicht bekannt ist, welche CPU der User gerade benutzt, sind Filter mit Vorsicht einzusetzen. Am Beispiel des Fade-Filters sei genannt, dass eine zeitgenaue Programmierung unmöglich ist, da die Abarbeitungsgeschwindigkeit des Filters neben den Zeitparametern auch von der CPU und sogar von der Version des Betriebssystems Windows abhängt (WXP rendert zügiger als W9x bei identischer CPU und Browserversion).

Die Synchronisierung des Filters z.B. mit Sound und ausgewählten Soundteilen ist aus o.g. Grund unmöglich, da es passieren kann, dass der Sound "schneller" ist. Mit anderen Worten: Der Fade-Filter braucht mehr Zeit, als seine Zeitparameter vorgeben, und ist somit nicht mehr synchron zu den Zeitparametern, die durch den Programmierer mit der Sounddauer abgestimmt sind. Daher sollte dann auf den Fade-Filter verzichtet und der Einsatz von transparenten Gif-Bildern oder einer in JScript animierten Bildfolge mit Soundsynchronisation z.B. per setTimeout() überdacht werden. Filter ist daher nur ein Dekorationselement.

Für ein Objekt, das einen Filter erhalten soll,

- muss per CSS mindestens eines der nachfolgenden Attribute kodiert sein: hihgt, width, position
wobei position absolute oder relative sein kann (falls nicht anderweitig festgelegt auf absolute).
- muss die Eigenschaft auf "tb-rl" oder Eigenschaft auf true gesetzt sein.



Folgende Objekte können **keinen** Filter bekommen:

- embed
- applet
- select im Formular
- option im Formular
- tr, thead, tbody,tfoot

Eine Zuordnung von Filtern zu den HTML-Elementen ist weiter unten zu sehen.

Event ist onfilterchange.

dokumentbezogener Filter:

Variante 1 per Meta-Tag

möglich für

- Page-Enter
- Page-Exit
- Site-Enter
- Site-Exit.

Beispiele

```
<META http-equiv="Page-Enter" CONTENT="progid:DXImageTransform.Microsoft.Blinds(Duration=4)" >
<META http-equiv="Page-Exit"
CONTENT="progid:DXImageTransform.Microsoft.Slide(Duration=2.500,slidestyle='HIDE')">
```

Variante 2 per HTML-Element-Tag mit STYLE-Attribut

BODY-Objekt kann Filter bekommen.

Beispiel

```
<BODY MARGINHEIGHT="0" TOPMARGIN="0" LEFTMARGIN="0" >
  <DIV STYLE="width:100%;height:100%;
    filter: progid:DXImageTransform.Microsoft.gradient
    (startColorstr=#550000FF, endColorstr=#55FFFF00)"
  >
  .....
  </DIV>
</BODY>
```

Hinweise zum Kodieren von Filtern:

Wenn mehrere Filter für ein Objekt kodiert werden, so entspricht ist die Abarbeitungsfolge der der Filterfolge laut Kodierung.

Filter immer als letzte Eigenschaft des Objektes kodieren

Für die Ausführung eines Filters muss das Objekt sichtbar sein (.style.visibility).

ELEMENT in der Tag-Anweisung ist durch ein konkretes Tag zu ersetzen.

object in der Script-Anweisung ist durch die konkrete Referenz auf ein Objekt zu ersetzen.

Syntax im IE 4.0 <Element STYLE="filter:filtername(Kette)" >

Syntax im IE 5.5 <Element STYLE="filter: progid:DXImageTransform.Microsoft.filtername(Kette)" >

Kette	String mit Liste aus den Eigenschaften des jeweiligen Filters
filtername	laut Filterart

Nachfolgende Filterbeschreibungen verwenden den Syntax ab IE 5.5.

Der Syntax ab IE 5.5 macht deutlich, wie Filter realisiert werden: Als windows-eigene Komponente.

Filterreferenz per Collection filters:

Beispiel

```
function setfilter()
{
    myimage.filters.item('DXImageTransform.Microsoft.alpha').opacity = document.forms(0).opacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').finishOpacity = document.forms(0).finishOpacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').style = document.forms(0).setstyle.value
}
```

Filterbezeichner vom IE 4.x und ab IE 5.5:

<u>4.0 filter name</u>	<u>Alternate 5.5 implementation</u>
alpha	DXImageTransform.Microsoft.Alpha
BlendTrans	DXImageTransform.Microsoft.Fade
blur	DXImageTransform.Microsoft.MotionBlur
chroma	DXImageTransform.Microsoft.Chroma
dropShadow	DXImageTransform.Microsoft.DropShadow
FlipH	DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)
FlipV	DXImageTransform.Microsoft.BasicImage(mirror=1)
glow	DXImageTransform.Microsoft.Glow



Gray	DXImageTransform.Microsoft.BasicImage(grayscale=1)
Invert	DXImageTransform.Microsoft.BasicImage(invert=1)
light	DXImageTransform.Microsoft.Light
mask	DXImageTransform.Microsoft.MaskFilter
shadow	DXImageTransform.Microsoft.Shadow
wave	DXImageTransform.Microsoft.Wave
Xray	DXImageTransform.Microsoft.BasicImage(xray=1)
RevealTrans	siehe unten

Filter RevealTrans bei IE 4.x und ab IE 5.5:

<u>numerischer Wert der Eigenschaft transition</u>	<u>alternative Kodierung ab IE 5.5</u>
0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Filter-Attribut des Style-Objekts:

.filter	Filter des Internetexplorer (siehe Objekt Filter)
Syntax:	<pre>object.style.filter = Kette [var Kette =] object.style.filter Kette Filter (siehe Objekt Filter)</pre>
lesen und schreiben	<pre>schreiben: immer überschreiben oder hinzufügen wird danach sofort geparkt: nachträgliches schreiben kostet Performance !!</pre>
Beispiel	<pre>Filter lesen <SCRIPT> alert(ID_Div.style.filter); // zeigt den String an // "filter:progid:DXImageTransform.Microsoft.Wave(strength=100) // progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)" </SCRIPT> <DIV ID="ID_Div" STYLE="height:50;width:50; filter:progid:DXImageTransform.Microsoft.Wave(strength=100) progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)" > Test-Text </DIV></pre>
Beispiel	<pre>Filter schreiben <DIV ID="ID_Div" STYLE="height:50;width:50;filter: progid:DXImageTransform.Microsoft.Wave(strength=100) progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)" > Test-Text </DIV> <SCRIPT> // hinzufügen ID_Div.style.filter += "progid:DXImageTransform.Microsoft.Iris(irisstyle='STAR',duration=4)"</pre>



</SCRIPT>

4.3.2.2.4.2.2.5.2. filters Collection des Internet Explorer

Feld aller Filter vom Objekt

Syntax:

```
[ var ZeigerAufFeld = ] object.filters
[ var ZeigerAufFeldElement = ] object.filters[Index, SubIndex]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes (analog zu NAME und ID-Attribut)
muss in [] kodiert sein		
SubIndex	optional	Integer Unterindex also Unterelement eines Elementes nur kodieren wenn Index ein String ist
ZeigerAufFeldElement		ist null, wenn Feldelement nicht vorhanden

Beispiel 1

```
<IMG SRC="test.jpg"
STYLE="filter: progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)
progid:DXImageTransform.Microsoft.Alpha(opacity=50)
progid:DXImageTransform.Microsoft.Blur(strength=10); position: relative"
>
<SCRIPT LANGUAGE="JavaScript">
    sample.filters.item(0).enabled = 1 // Numerischer Index
    sample.filters.item("DXImageTransform.Microsoft.Alpha").enabled = 0 // Namen-Index
</SCRIPT>
```

Beispiel 2

```
function setfilter()
{
    myimage.filters.item('DXImageTransform.Microsoft.alpha').opacity = document.forms(0).opacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').finishOpacity = document.forms(0).finishOpacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').style = document.forms(0).setstyle.value
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.2.2.5.3. Filtereigenschaften

Nachfolgende Eigenschaften werden nicht von allen Filtern benötigt. Eigenschaften werden z.T. mit identischen Bezeichnern anders wertmässig belegt (je nach Filter).

.add	Überlagerung des Filters über das Originalbild
.bands	Anzahl der Streifen
.Bias	Prozentsatz der Farbwerterhöhung je höher der Bildkontrast um so geringer der sichtbare Effekt
.color	Farbe
.colorSpace	Pfad der *.icm-Datei
.direction	Filterrichtung nicht Filter Blinds und CheckerBoard

Beispiel

```
<SCRIPT>
function Setze()
{
    with (window.event.srcElement.filters[0])
    {
        if (strength < 100)
        {
            strength += 1;
            direction += 45;
        }
    }
}
```



	<pre> } </SCRIPT> </pre>
.Direction	Richtung
.duration	nur Filter Blinds und CheckerBoard Dauer der Animation des Filters
Beispiel	<pre> <SCRIPT LANGUAGE=JavaScript> var StartBild = "/graphics/test1.jpg"; var EndeBild = "/graphics/test2.jpg"; function FilterAusfuehren() { ID_Img.filters.item(0).Apply(); ID_Img.src = EndeBild; EndeBild = StartBild; // neues Endebild StartBild = ID_Img.src; // altes Endebild ID_Img.filters.item(0).Play(); } </SCRIPT>
 <INPUT TYPE="button" value="Start Transition" onClick="FilterAusfuehren()"> </pre>
.Dx	X-Komponente des Vektor der linearen Transformation
.Dy	Y-Komponente des Vektor der linearen Transformation
.enabled	wird ignoriert wenn Eigenschaft .SizingMethod mit Wert "auto expand" Filter-Anwendbarkeit
Beispiel	<pre> </pre>
.EndColor	Ende-Deckungskraft als Integer
.EndColorStr	Ende-Deckungskraft als String
.FilterType	Filtertyp
.finishOpacity	Deckungskraft, mit der Filter endet
.finishX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft endet
.finishY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft endet
.freq	Anzahl der Wellen
.function	Nummer der Komposition
.gradientSize	Faktor der Objektbreite als Gradientbreite
.GradientType	Orientierung des Gradienten
.grayScale	Grauskala ein/aus
Beispiel	<pre> <DIV ID="ID_Div" STYLE="position:absolute; left:270px;" > </DIV> <BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(grayScale=1)'"> Gray </BUTTON>
 <BUTTON onclick="ID_Div.style.filter=''"> Clear Filter </BUTTON> </pre>
.gridSizeX	Anzahl Spalten des Gitters
.gridSizeY	Anzahl Zeilen des Gitters
.intent	Farbverwendungsart
.Invert	Komplementärfarben ein/aus
Beispiel	<pre> <DIV ID="ID_Div" STYLE="position:absolute; left:270px;" > </DIV> <BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(invert=1)'"> </pre>




```

        Invert
    </BUTTON>
    <BR>
    <BUTTON onclick="ID_Div.style.filter="">
        Clear Filter
    </BUTTON><BR>
    .irisStyle        Schärfe des Filters
    .lightStrength    prozentuale Differenz der Licht-Intensität zwischen Wellengipfel und Wellentäler
    .M11              Matrixfeld M11
    .M12              Matrixfeld M12
    .M21              Matrixfeld M21
    .M22              Matrixfeld M22
    .makeShadow       Schatten ein/aus

```

Beispiel

```

<DIV STYLE="position:absolute; left:70px; filter:
    progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
        ShadowOpacity=1.0)"
>
    <IMG SRC="/test.gif" >
</DIV>

```

.Mask Transparenzänderung auf Wert laut Eigenschaft .MaskColor ein/aus

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>
.MaskColor    Farbmaske

```

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>

```

.maxSquare Maximale Anzahl der Pixels im Quadrat
 .mirror Rervers ein/aus

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mirror=1)'">
    Rotate 270 degrees
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>

```

.motion Bewegungsrichtung
 nicht Filter GradientWipe und Strips
 .motion Art der Bewegung
 nur Filter GradientWipe



.motion Ecke
 nur Filter Strips
 .offX horizontaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .offY vertikaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .opacity Deckungskraft mit der der Filter startet
 nicht Filter BasicImage
 .opacity Deckungskraft- bz.w Transparenz-Niveau (Opacity-Niveau)
 nur Filter BasicImage

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
  <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(opacity=.2)'">
  Opaque
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter=''">Clear Filter</BUTTON>
Klick Opaque-Button für Transparenz
```

.orientation Orientierung
 .overlap Anteil der Überlappungs-Dauer am Wert laut Eigenschaft .duration

Beispiel

wenn .overlap auf 0.4 und .duration auf 10 Sekunden
 dann 40 % der 10 Sekunden findet überlappen statt = 4 Sekunden
 60 % der 10 Sekunden ohne überlappen = 6 Sekunden, also
 30 % der 10 Sekunden nur Fadeout = 3 Sekunden
 30 % der 10 Sekunden nur Fadein = 3 Sekunden
 also Ablauf:
 Sekunde 1-3 Fadeout aber nicht komplett
 Sekunde 4-7 Fadeout komplett UND Fadein beginnt = Overlap
 Sekunde 8-10 Fadein komplett
 Prozentualer Umfang der Anzeige mit dem der Filter enden soll

.Percent

Beispiel

```
<SCRIPT>
function FilterSetzen()
{
  ID_Div.filters[0].Apply();

  // Achtung: nach Applay() kodierte Veränderungen vom DIV werden ERST
  // mit der Abarbeitung der Methode .play() sichtbar
  ID_Div.style.backgroundColor="blue";
  ID_Div.filters[0].percent=50;
  ID_Div.filters[0].enabled=true;
}
</SCRIPT>
<BUTTON onclick="FilterSetzen(); onclick=''">FilterSetzen</BUTTON>
<BR>
<DIV ID="ID_Div"
  STYLE="height:250px; width:250px; background-color: gold;
  filter:progid:DXImageTransform.Microsoft.checkerboard(duration=5, transition=7);"
>
</DIV>
```

.phase prozentuale Phasenverschiebung der Wellen,
 also Verschiebung der Wellenüberlagerung als Prozentanteil des Wellenzyklus
 .pixelRadius Radius des Blur Filter-Raumes

Beispiel

```
<DIV STYLE="position:absolute; left:70px; filter:
  progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
  ShadowOpacity=1.0)"
>
  <IMG SRC="/test.gif" >
</DIV>
```

.positive Schattentransparenz
 .Rotation Faktor der Rotation in 90-Grad-Schritten

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
  <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(rotation=3)'">
  Rotate 270 degrees
</BUTTON>
```



```

<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>
.shadowOpacity      Schatten-Deckungskraft
                    wirkt nur wenn Eigenschaft makeShadow auf true gesetzt ist
    Beispiel
<DIV STYLE="position:absolute; left:70px; filter:
    progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
        ShadowOpacity=1.0)"
>
    <IMG SRC="/test.gif" >
</DIV>
.sizingMethod      Art der Anpassung
                  nicht Filter Matrix
    Beispiel
<SCRIPT>
    function FilterAendern(ArtDerAnpassung)
    { ID_Div.filters(0).sizingMethod= ArtDerAnpassung; }
</SCRIPT>
<DIV ID="ID_Div"
    STYLE="position:absolute; left:140px; height:150; width:200;
        filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
            SRC='test.jpg', sizingMethod='scale');"
>
</DIV>
<BUTTON onclick=" FilterAendern('image');">kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>
.SizingMethod      Container-Resize ein/aus
                  nur Filter Matrix
.slideStyle        Art
    Beispiel
<DIV STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Slide(
        duration=3, bands='8', slideStyle='PUSH');"
>
.spokes            Anzahl Speichen
.squaresX          Anzahl der Spalten
    Beispiel
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else // 0 entspricht false
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
        duration=5, direction='left', squaresX=4);">
</DIV>
.squaresY          Anzahl der Zeilen
    Beispiel
<SCRIPT>

```



```

var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
        duration=5, direction='left', squaresY=4);">

.SRC      Url des Image
Beispiel

<SCRIPT>
    function FilterAendern(ArtDerAnpassung)
    { ID_Div.filters(0).sizingMethod= ArtDerAnpassung; }
</SCRIPT>
<DIV ID="ID_Div"
    STYLE="position:absolute; left:140px; height:150; width:200;
        filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
            SRC='test.jpg', sizingMethod='scale');"
    >
</DIV>
<BUTTON onclick=" FilterAendern('image');"> kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>

.StartColor      Start-Deckungskraft als Integer
.StartColorStr    Start-Deckungskraft als String
.startX           Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft startet
.startY           Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft startet
.status           Status der Filteranimation
.strength         Ausdehnung des Glühens
Beispiel

<SCRIPT>
    function FilterSetzen()
    {
        with (window.event.srcElement.filters[0])
        {
            if (strength < 200)
            {
                strength += 1;
                direction += 45;
            }
        }
    }
</SCRIPT>
<IMG SRC="test.gif"
    STYLE="filter:progid:DXImageTransform.Microsoft.motionBlur(STRENGTH=1, DIRECTION=0)"
    onfilterchange="FilterSetzen()"
    >

.stretchStyle     Art
Beispiel

<DIV STYLE="height:250px; width:250px; background-color: gold;

```



```
filter:progid:DXImageTransform.Microsoft.Stretch(duration=2, stretchStyle='PUSH');"
```

>	Art der Deckungskraft
.style	Filternummer
.transition	
<u>numerischer Wert der Eigenschaft transition</u>	<u>alternative Kodierung ab IE 5.5</u>
0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Beispiel

```
<SCRIPT>
function go()
{
    ID_Span.filters[0].Apply();

    if (ID_Span.style.visibility == "visible")
    {
        ID_Span.style.visibility = "hidden";
        ID_Span.filters.revealTrans.transition=2;
    }
    else
    {
        ID_Span.style.visibility = "visible";
        ID_Span.filters[0].transition=3;
    }

    ID_Span.filters[0].Play();
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Play Transition" onClick="go();" ></INPUT>
<SPAN ID="ID_Span" STYLE="position:absolute; visibility:visible;
    filter:revealTrans(DURATION=2, TRANSITION=3);
    width:300; height:300; background-color: lightgreen"
>
    <CENTER>
        <DIV STYLE="background-color:red;height=100;width:100;
            position:relative;top:100
            "
        >
        </DIV>
    </CENTER>
</SPAN>
.WipeStyle    Richtung
               nicht Filter RadialWipe
.wipeStyle    Style des Wischens
               nur Filter RadialWipe
.Xray         Grauskale aus Mittelwert vom Rot- und Grünanteil ein/aus
```

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; color:tan;" >
    <IMG SRC="/test.gif" >
```



```

</DIV>
<BUTTON onclick="ID_Div.style.filter= 'progid:DXImageTransform.Microsoft.BasicImage(XRay=1)'">
    Show Xray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>

```

4.3.2.2.4.2.5.4. Filtermethoden

.addAmbient()	indirektes Umgebungslicht (Ambiente)
.addCone()	direktes 3D-Licht als Lichttunnel oder Spot
.addPoint()	D-Lichtpunkt als Lichtquelle, die in alle Richtungen strahlt
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart die Methode .play() benutzen (sichtbarmachen der Filtereinstellungen)
.changeColor()	Lichtfarbe nachträglich ändern
.changeStrength()	Lichtintensität nachträglich ändern
.clear()	alle Lichtquellen löschen
.moveLight()	Focus des Lichtkegels bzw. Lichtpunkt bewegen
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.5.5. Alpha Filter

Deckungskraft eines sichtbaren Objektes

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Alpha(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Alpha(Kette)"
	Kette String mit Eigenschaftenliste

Beispiel für transparentes INPUT:

```

<STYLE>
    INPUT.aFilter { filter:progid:DXImageTransform.Microsoft.Alpha(opacity=50);}
</STYLE>
<INPUT TYPE="button" VALUE="Button" CLASS="aFilter">

```

Beispiel für transparentes Bild auf einem Hintergrundbild:

```

<STYLE>
    IMG.filter_name { filter:Alpha(opacity=y);}
</STYLE>
<BODY BACKGROUND="hintergrund_bild.gif">
    <IMG SRC="bild.gif" .... CLASS="filter_name">
</BODY>

```

Beispiel für Uhr:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
//          24-Stunden Digitaluhr mit Spiegelbild nuf für IE
//
// *****
//
//          vom Programmierer zu belegende Variablen
//
// *****
var Uhr_Frequenz=960;                                // Bitte an die echte Genauigkeit anpassen auf Sekundentakt

var Uhr_Breite           = 10;                        // in Pixel
var Uhr_Hoehe            = 10;                        // in Pixel
var Uhr_AbstandVomFensterRandOben = 10;              // in Pixel
var Uhr_AbstandVomFensterRandLinks = 50;             // in Pixel

var Uhr_UngespiegeltFontFamily      = "Comic Sans Ms";
var Uhr_UngespiegeltFontSize        = 14;             // in pt
var Uhr_UngespiegeltFontStyle       = "italic";
var Uhr_UngespiegeltFontWeigth      = "bold";

```



```

var Uhr_UngespiegeltFontFarbe      = "#00008D";

var Uhr_GespiegeltFontFamily      = "Comic Sans Ms";
var Uhr_GespiegeltFontSize        = 14;                // in pt
var Uhr_GespiegeltFontStyle       = "italic";
var Uhr_GespiegeltFontWeigth     = "bold";
var Uhr_GespiegeltFontFarbe      = "#00008D";
var Uhr_GespiegeltDeckungskraft   = 11;                // je höher um so stärker die Farbe denkend
// je kleiner um so blasser die Farbe

var Uhr_GespiegeltAbstandVonUngespiegelt = Uhr_UngespiegeltFontSize + 6;

var LeeresBildUrl="1x1tran.gif"                // muss 1x1 Pixel sein, ist transparent

// *****
//
//      nachfolgenden Code nicht verändern
//
// *****
// Dokument mit leeren Bild instanzieren
//      Instanzierung erfolgt normalerweise ERST mit Lesen des BOD>-Tags, aber
//      das Dokument muss spätestens zur Ermittlung der Fensterdimensionen
//      per Javascript instanziiert sein, also auch in Javascript.
//      ACHTUNG: Der GESAMTE BODY-Teil des Dokumentes wird damit ignoriert und MUSS leer bleiben !

document.write('<IMG SRC="' + LeeresBildUrl + '" HEIGHT=1 WIDTH=1>');

//      Prüfung auf Browsertyp
//      Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

var TimeoutID;

function UhrAnzeigen()
{
    var DatumObjekt    = new Date();
    var Stunde         = DatumObjekt.getHours();
    var Minute         = DatumObjekt.getMinutes();
    var Sekunde        = DatumObjekt.getSeconds();

    if (Stunde < 10 )
    {Stunde="0" + Stunde;}

    if (Minute <10 )
    {Minute="0" + Minute;}

    if (Sekunde <10 )
    {Sekunde="0" + Sekunde;}

    DIV_ID_Uhr_ungespiegelt.innerHTML = Stunde + ":" + Minute + ":" +Sekunde;
    DIV_ID_Uhr_gespiegelt.innerHTML  = Stunde + ":" + Minute + ":" +Sekunde;

    TimeoutID=setTimeout('UhrAnzeigen()',Uhr_Frequenz);
}

if (ie)
{
    document.write(    '<DIV ID="DIV_ID_Uhr_ungespiegelt"'
        +      ' STYLE="position:absolute;'
        +      'width:'      + Uhr_Breite      + 'px;'
        +      'height:'    + Uhr_Hoehe      + 'px;'
        +      'top:'       + Uhr_AbstandVomFensterRandOben + 'px;'
        +      'left:'      + Uhr_AbstandVomFensterRandLinks + 'px;'
        +      'font-family:' + Uhr_UngespiegeltFontFamily + ';'
        +      'font-size:'  + Uhr_UngespiegeltFontSize + 'pt;'
        +      'font-style:' + Uhr_UngespiegeltFontStyle + ';'
        +      'font-weight:' + Uhr_UngespiegeltFontWeigth + ';'
        +      'color:'     + Uhr_UngespiegeltFontFarbe + ';'
        +      ""
        + '>'
        + '</DIV>'

```



```

    );

    document.write(
        '<DIV ID="DIV_ID_Uhr_gespiegelt"'
        + ' STYLE="position:absolute;'
        + ' width:' + Uhr_Breite + 'px;'
        + ' height:' + Uhr_Hoehe + 'px;'
        + ' top:'
        + (Uhr_AbstandVomFensterRandOben + Uhr_GespiegeltAbstandVonUngespiegelt)
        + 'px;'
        + ' left:' + Uhr_AbstandVomFensterRandLinks + 'px;'
        + ' font-family:' + Uhr_GespiegeltFontFamily + ';'
        + ' font-size:' + Uhr_GespiegeltFontSize + 'pt;'
        + ' font-style:' + Uhr_GespiegeltFontStyle + ';'
        + ' font-weight:' + Uhr_GespiegeltFontWeight + ';'
        + ' color:' + Uhr_GespiegeltFontFarbe + ';'
        + ' filter:flipv() alpha(opacity=' + Uhr_GespiegeltDeckungskraft + ');'
        + '";'
        + '>'
        + '</DIV>'
    );

    UhrAnzeigen();
}
//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.finishOpacity	Deckungskraft, mit der Filter endet
.finishX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft endet
.finishY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft endet
.opacity	Deckungskraft mit der der Filter startet
.startX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft startet
.startY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft startet
.style	Art der Deckungskraft

Methoden:

keine

4.3.2.2.4.2.2.5.6. AlphaImageLoader Filter

Imageanpassung an Container-Dimension

Syntax:

```

HTML    <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.AlphaImageLoader(Kette)"

```

Kette String mit Eigenschaftenliste

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.SRC	Url des Image
.sizingMethod	Art der Anpassung

Methoden:

keine

4.3.2.2.4.2.2.5.7. Barn Filter

Scheunentür-Animation (öffnen/schliessen wie Türen)

ab IE 5.5

Syntax:

```

HTML    <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Barn(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Barn(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        // Filter init
        ID_Div.filters[0].Apply();
    }

```




```

// Filter ändern
if (Flag) // true entspricht numerisch 1
{
    Flag = 0;
    ID_Div.style.backgroundColor="gold";
}
else // false entspricht numerisch 0
{
    Flag = 1;
    ID_Div.style.backgroundColor="blue";
}

// Filter animieren
ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Barn(
    duration=2, motion='out', orientation='vertical');"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.motion	Bewegungsrichtung
.orientation	Orientierung
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.8. BasicImage Filter

Farbverläufe, Bildrotation, Deckkraft

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.BasicImage(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.BasicImage(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```

<STYLE>
    button { width:250px; }
</STYLE>
<DIV ID="ID_Div" oDiv" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mirror=1)'">
    Mirror
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(invert=1)'">
    Invert
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(grayScale=1)'">
    GrayScale
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(xray=1)'">
    X-Ray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mask=1,

```



```

                                maskColor=255)"">
                                Blue MaskColor
        </BUTTON>
        <BR>
        <BUTTON onclick=" ID_Div.style.filter="">
                                Clear Filter
        </BUTTON>
        <BR>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.grayScale	Grauskala ein/aus
.Invert	Komplementarfarben ein/aus
.Mask	Transparenzänderung auf Wert laut Eigenschaft .MaskColor ein/aus
.MaskColor	Farbmaske
.mirror	Rervers ein/aus
.opacity	Deckungskraft- bz.w Transparenz-Niveau (Opacity-Niveau)
.Rotation	Achtung: nur die Eigenschaft vom Filter BasicImage
.Xray	Faktor der Rotation in 90-Grad-Schritten
	Grauskala aus Mittelwert vom Rot- und Grünanteil ein/aus

Methoden:

keine

4.3.2.2.4.2.2.5.9. BlendTrans Filter

Blenden-Filter (Fading) für ein sichtbares Objekt

ab IE 4.x

Syntax:

```

HTML    <ELEMENT STYLE="filter:BlendTrans(Kette)" ... >
Scripting object.style.filter ="BlendTrans(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<HEAD>
<SCRIPT>
    function fadeOut()
    {
        ID_Div.style.filter="blendTrans(duration=2)";
        // prüfen ob Filter nicht gerade animiert
        if (ID_Div.filters.blendTrans.status != 2)
        {
            // Filter animiert nicht
            ID_Div.filters.blendTrans.apply();
            ID_Div.style.visibility="hidden";
            ID_Div.filters.blendTrans.play();
        }
    }

    function fadeIn()
    {
        ID_Div.style.filter="blendTrans(duration=2)";
        // prüfen ob Filter nicht gerade animiert
        if (ID_Div.filters.blendTrans.status != 2)
        {
            // Filter animiert nicht
            ID_Div.filters.blendTrans.apply();
            ID_Div.style.visibility="visible";
            ID_Div.filters.blendTrans.play();
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <!-- DIV muss mit Layout deklariert sein z.B. Breite -->
    <DIV ID="ID_Div" STYLE="width: 200">
        Text fuer faded in und out.

    </DIV>
    <P>
        <BUTTON onclick="fadeOut()">Fade Text Out</BUTTON>
        <BUTTON onclick="fadeIn()">Fade Text In</BUTTON>

    </P>
</BODY>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
-----------	---------------------------------



.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
Methoden:	
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.5.10. Blinds Filter

Blendenfilter (Blenden öffnen/schliessen)

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Blinds(Kette)" ... >
Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Blinds(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var Flag = 0;
function Aendern()
{
    // Filter init
    ID_Div.filters[0].Apply();
    // Anzahl der Streifen zufällig ermitteln
    ID_Div.filters[0].bands = Math.random()*12 + 3;
    // Filter ändern
    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }
    // Filter starten
    ID_Div.filters[0].Play(duration=3);
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern</BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Blinds(direction='down');">
</DIV>
```

Eigenschaften:

.bands	Anzahl der Streifen
.Direction	Richtung nur Filter Blinds und CheckerBoard
.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.5.11. Blur Filter

Filter für Objekt, wenn es keinen Focus hat



ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Blur(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Blur(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT LANGUAGE="JScript">
    function Aendern(ButtonZeiger)
    {
        if (ID_Div.filters(0).enabled)
        {
            ID_Div.filters(0).enabled='false';
            ButtonZeiger.innerText='blur';
        }
        else
        {
            ID_Div.filters(0).enabled='true';
            ButtonZeiger.innerText='normal';
        }
    }
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
    progid:DXImageTransform.Microsoft.blur(pixelradius=3, enabled='false')">
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern(this)">blur</BUTTON><BR>
```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.makeShadow	Schatten ein/aus
.pixelRadius	Radius des Blurfilter-Raumes
.shadowOpacity	Schatten-Deckungskraft
	wirkt nur wenn Eigenschaft makeShadow auf true gesetzt ist

Methoden:

keine

4.3.2.2.4.2.2.5.12. CheckerBoard Filter

Schachbrettfiter

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.CheckerBoard(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.CheckerBoard(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else // 0 entspricht false
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
        duration=5, direction='left');">
```



</DIV>

Eigenschaften:

.Direction	Richtung nur Filter Blinds und CheckerBoard
.squaresX	Anzahl der Spalten
.squaresY	Anzahl der Zeilen

Methoden:

keine

4.3.2.2.4.2.2.5.13. Chroma Filter

Anzeige einer Farbe aus der Farbtiefe als transparent

wenn Farbe bereits transparent so durch Filter als deckend erzeugt

bei JPG-Files kaum wirksam !!!!!, da Farbtiefe reduziert ist

bei Rand-geglätteten Objekten nicht wirksam, da Glättung durch Anpassung an umgebende Pixel-Farbwerte erfolgt
(Verschwommenheit, Weichmachen)

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Chroma(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Chroma(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<BUTTON onclick=" ID_Div.filters.item('DXImageTransform.Microsoft.Chroma').color = 'gray';">
  grau zu blau
</BUTTON>
<BR>
<BUTTON onclick=" ID_Div.filters.item('DXImageTransform.Microsoft.Chroma').color = 'blue';">
  blau zu grau
</BUTTON>
<DIV ID="ID_Div" STYLE="position:absolute; top:100; left:10; width:240; height:100;
  filter:progid:DXImageTransform.Microsoft.Chroma(color='yellow'),
  progid:DXImageTransform.Microsoft.Chroma(color='red')">
  >
  <BR>
  <SPAN STYLE="position:absolute; top:50; left:10; width:240; height:60;color:gray;">
    grau
  </SPAN>
  <BR>
  <SPAN STYLE="position:absolute; top:70; left:10; width:240; height:60; color:blue;">
    blau
  </SPAN>
</DIV>
```

Eigenschaften:

.color	Farbe
.enabled	Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.14. Compositor Filter

Farbkombination

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Compositor(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Compositor(Kette)"

Kette String mit Eigenschaftenliste

Ablauf: Funktion wählen

Input A erzeugen Image
per Methode .apply()
 Input B erzeugen Image
 Eigenschaften des Objektes (eventuelle von Kindern des Objektes) verändern
 z.B. visibility, innerText, backgroundColor, border

Methode play() aufrufen

Eigenschaften:

.function	Nummer der Komposition
-----------	------------------------

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen



4.3.2.2.4.2.2.5.15. DropShadow Filter

Anzeige eines Konturenrahmens mit verschiedener Schattenrichtung

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.DropShadow(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.DropShadow(Kette)"

Kette String mit Eigenschaftenliste
 Beispiel

```
<SCRIPT>
function Negieren()
{
    var Status = ID_Div.filters.item('DXImageTransform.Microsoft.dropshadow').enabled;
    ID_Div.filters.item('DXImageTransform.Microsoft.dropshadow').enabled = !Status;
}
</SCRIPT>
<DIV ID="ID_Div"
STYLE="position:absolute; top:50; left:10; width:240;
height:160; padding:10px; font:bold 13pt verdana;
filter:progid:DXImageTransform.Microsoft.dropshadow(OffX=5, OffY=5,
Color='gray', Positive='true')
onclick="Negieren();"
>
Test-Text
</DIV>
```

Eigenschaften:

.color Farbe
 .enabled Filter-Anwendbarkeit
 .offX horizontaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .offY vertikaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .positive Schattentransparenz

Methoden:

keine

4.3.2.2.4.2.2.5.16. Emboss Filter

emossed Texttur mit Grauskala

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Emboss(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Emboss(Kette)"

Kette String mit Eigenschaftenliste
 Beispiel

```
<SCRIPT>
function Aendern(ButtonZeiger)
{
    if (ID_Div.filters(0).enabled)
    {
        ID_Div.filters(0).enabled='false';
        ButtonZeiger.innerText='embossed';
    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='normal';
    }
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
progid:DXImageTransform.Microsoft.emboss(enabled='false')
>
<IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern(this)"> Aendern </BUTTON>
```

Eigenschaften:

.Bias Prozentsatz der Farberhöhung
 je höher der Bildkontrast um so geringer der sichtbare Effekt
 .enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.17. Engrave Filter

engraved Texttur mit Grauskala

(TWS) Microsoft JScript für den Hobby-Programmierer



ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Engrave(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Engrave(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
function Aendern (ButtonZeiger)
{
    if (ID_Div.filters(0).enabled)
    {
        ID_Div.filters(0).enabled='false';
        ButtonZeiger.innerText='Make Engraved';
    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='Make Normal';
    }
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
progid:DXImageTransform.Microsoft.engrave(enabled='false')"
>
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern(this)"> Aendern </BUTTON>
```

Eigenschaften:

.Bias Prozensatz der Farbwerthöhung
 je höher der Bildkontrast um so geringer der sichtbare Effekt
 .enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.18. Fade Filter

Fadeout und FadeIn mit überlappen

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Fade(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Fade(Kette)"

Kette String mit Eigenschaftenliste

Beispiel für Fade eines DIV:

```
<SCRIPT>
var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON><BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Fade(duration=2);"
>
```



</DIV>

Beispiel für Aus- und Einblende eines Bildes:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var Sichtbar=true;    // DIV ist natürlich nach dem Dokument-Laden sichtbar
    var StandardFarbe="green";

    function Blenden()
    {
        // Fade zurücksetzen
        DIV_ID.filters[0].Apply();

        if (Sichtbar)
        {
            Sichtbar=false;
            DIV_ID.style.visibility="hidden";
        }
        else
        {
            Sichtbar=true;
            DIV_ID.style.visibility="visible";
            DIV_ID.style.backgroundColor=StandardFarbe;
        }

        // Fade starten
        DIV_ID.filters[0].Play();
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="Blenden()">
    Aus- und Einblenden
</BUTTON>
<BR>
<BR>
<DIV    ID="DIV_ID"
    STYLE="height:250px;width:250px;background-color:red;
        filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile !!
>
    <IMG SRC="Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
</DIV>
</BODY>
</HTML>
```

Beispiel für Aus- und Einblende von Bildern mit Bildwechsel:

Variante 1

Es werden zwei DIV an absoluter Position überlagert. Jedes DIV hat einen eigenen Inhalt, hier im Beispiel sein eigenes Bild. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der beiden Bilder erreicht. Pro Bild wird ein eigenes DIV erzeugt.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var DIV_ID1_Sichtbar=true;    // DIV_ID1 ist nach dem Dokument-Laden sichtbar
                                   // DIV_ID2 aber nicht

    function BildWechsel()
    {
        // Fade zurücksetzen
        DIV_ID1.filters[0].Apply;
        DIV_ID2.filters[0].Apply;
```




```

        if (DIV_ID1_Sichtbar)
        {
            DIV_ID1_Sichtbar = false;
            DIV_ID1.style.visibility="hidden";
            DIV_ID2.style.visibility="visible";
        }
        else
        {
            DIV_ID1_Sichtbar = true;
            DIV_ID1.style.visibility="visible";
            DIV_ID2.style.visibility="hidden";
        }
        // Fade starten
        DIV_ID1.filters[0].Play();
        DIV_ID2.filters[0].Play();
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
    <BUTTON onclick="BildWechsel()">
        Bild wechseln mit Aus- und Einblenden
    </BUTTON>
    <BR>
    <BR>
    <DIV ID="DIV_ID1"
        STYLE="height:250px;position:absolute;top:50;width:250px;
            background-color:red; visibility:visible
            filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile
    >
        <IMG SRC="DIV_ID1_Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
    </DIV>

    <DIV ID="DIV_ID2"
        STYLE="height:250px;position:absolute;top:50;width:250px;
            background-color:red; visibility:hidden
            filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile
    >
        <IMG SRC="DIV_ID2_Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
    </DIV>
</BODY>
</HTML>

```

Variante 2

Es wird der innere DIV-Bereich zwischen <DIV> und </DIV> ersetzt und damit je ein Bild dargestellt. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der Bilder erreicht.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var BildUrlFeld=Array
    (
        // hier beliebig viele Bilder eintragen
        "test1.jpg",
        "test2.jpg",
        "test3.jpg"
    );

    var BildUrlFeld_Laenge= BildUrlFeld.length;

    //      fuer alle Bilder die identischen Dimensionen !!!
    var BildHoehe=444;
    var BildBreite=640;

    // Zeigerfeld zum Vorladen der Grafiken
    var BildZeigerFeld = Array();

```



```

var Index=1;           // Starten mit test2.jpg, da test1.jpg bereits angezeigt mit Laden des DIV

function BildWechsel()
{
    // Fade zurücksetzen
    DIV_ID.filters[0].Apply();

    // Bild im DIV anzeigen
    document.AktuellesBild.src= BildZeigerFeld[Index].src;

    // Fade starten
    DIV_ID.filters[0].Play();

    // nächstes Bild einstellen
    Index++;

    // Index korrigieren
    if (Index >= BildUrlFeld_Laenge)
    {Index=0;}
}

// nachfolgender Code wird mit dem Laden des HEAD-Teiles abgearbeitet

// Bildobjekte vorladen: allokalieren und Zeiger merken per Prototyping
for (i=0; i<= BildUrlFeld_Laenge; i++)
{
    // Image-Zeiger bilden als Feldeintrag
    BildZeigerFeld[i] = new Image();

    // Url dem Zeiger zuweisen und Bild somit vorladen
    BildZeigerFeld[i].src= BildUrlFeld[i];
}

// DIV erzeugen mit test1.gif als Startbild
//                               in Dimension aller Bilder
document.write(    '<DIV '
                  +   'ID="DIV_ID" '
                  +   'STYLE="height: ' + BildHoehe + 'px;width: ' + BildBreite + 'px;'
                  +   'filter:progid:DXImageTransform.Microsoft.Fade(duration=1);'
                  +   '""
                  +   '>\n'
                  );

document.write(    '<IMG NAME="AktuellesBild"'
                  +   'SRC="" + BildZeigerFeld[0].src + "'
                  +   'WIDTH=' + BildBreite + "'
                  +   'HEIGHT=' + BildHoehe
                  +   '>\n'
                  );

document.write(    '</DIV>\n');

// Button zum Bildwechsel anzeigen
document.write(    '<BUTTON onclick="BildWechsel()">'
                  +   'Nächstes Bild'
                  +   '</BUTTON>\n'
                  );

// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.overlap	Anteil der Überlappungs-Dauer am Wert laut Eigenschaft .duration
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)

.play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen

.stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.19. FlipH Filter

horizontale Objekt-Spiegelung

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:FlipH" ... >
Scripting object.style.filter ="FlipH"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.20. FlipV Filter

vertikale Objekt-Spiegelung

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:FlipV" ... >
Scripting object.style.filter ="FlipV"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.21. Glow Filter

Glühen an den Objekträndern

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Glow(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Glow(Kette)"

Kette String mit Eigenschaftenliste

Hinweise: Objekt mit Text ohne Hintergrundfarbe --> Textzeichen glühen
Objekt mit Hintergrundfarbe oder Image --> Objektränder glühen
Objekt mit Kind, das aber außerhalb des Elternobjektes liegt --> Kind glüht nicht !!

Beispiel

```
<STYLE>
  DIV.aFilter { filter: glow(Color=blue,Strength=5); width: 150;}
</STYLE>
<DIV CLASS="aFilter">gluehender Text</DIV>
```

Eigenschaften:

.color Farbe
.enabled Filter-Anwendbarkeit
.strength Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.22. Gradient Filter

Farbe zwischen Hintergrund und Objektfarbe

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Gradient(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Gradient(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>

function Aendern(ButtonZeiger)
{
  if (ID_Div.filters(0).enabled)
  {
    ID_Div.filters(0).enabled='false';
    ButtonZeiger.innerText='gradient';
  }
}
```



```

    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='normal';
    }
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="height:120px; color:green; filter:
    progid:DXImageTransform.Microsoft.gradient(enabled='false',
    startColorstr=#550000FF, endColorstr=#55FFFF00)"
>
    Test-Text
</DIV>
<BUTTON onclick="Aendern(this)"> Aendern </BUTTON>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.EndColor	Ende-Deckungskraft als Integer
.EndColorStr	Ende-Deckungskraft als String
.GradientType	Orientierung des Gradienten
.StartColor	Start-Deckungskraft als Integer
.StartColorStr	Start-Deckungskraft als String

Methoden:

keine

4.3.2.2.4.2.2.5.23. GradientWipe Filter

Gradientstreifen

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.GradientWipe(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.GradientWipe(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="orange";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON><BR><BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: orange;
    filter:progid:DXImageTransform.Microsoft.gradientWipe(
    duration=3, gradientsize=0.5);"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.gradientSize	Faktor der Objektbreite als Gradientbreite
.motion	Art der Bewegung
	nur Filter GradientWipe
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.WipeStyle	Richtung

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück



nach apply sind die Filtereinstellungen zu kodieren
 für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
 Animiert den Filter in seinen aktuellen Einstellungen
 benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
 .play()
 .stop() Stop der Filteranimation vor dem normalen Ende
 erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.24. Gray Filter

Eingrauen per Graustufenskala

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:Gray" ... >
 Scripting object.style.filter ="Gray"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.25. ICMFilter Filter

Farbkonvertierung auf Basis von Image Color Management Profilen (*.ICM-Dateien) von Hardware wie z.B. Printer etc..

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.ICMFilter(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.ICMFilter(Kette)"

Kette String mit Eigenschaftenliste

Eigenschaften:

.colorSpace Pfad der *.icm-Datei
 .intent Farbverwendungsart

Methoden:

keine

4.3.2.2.4.2.2.5.26. Inset Filter

Diagonalfilter

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Inset(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Inset(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
  var Flag = 0;

  function Aendern()
  {
    ID_Div.filters[0].Apply();

    if (Flag)
    {
      Flag = 0;
      ID_Div.style.backgroundColor="gold";
    }
    else
    {
      Flag = 1;
      ID_Div.style.backgroundColor="blue";
    }
    ID_Div.filters[0].Play(duration=2);
  }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
  filter:progid:DXImageTransform.Microsoft.Inset( );"
>
</DIV>
```

Eigenschaften:

.duration Dauer der Animation des Filters
 .enabled Filter-Anwendbarkeit



.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
Methoden:	
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.27. Invert Filter

Invertieren (Negativ)

ab IE 4.x

Syntax:

HTML	<ELEMENT STYLE="filter:Invert" ... >
Scripting	object.style.filter = "Invert"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.28. Iris Filter

Kamera-Optik-Filter (wie die Pupille)

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Iris(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Iris(Kette)"

Beispiel Kette String mit Eigenschaftenliste

```

<SCRIPT>
var FeldDerIrisStyles = new Array();
FeldDerIrisStyles = ['DIAMOND','CIRCLE','CROSS','PLUS','SQUARE','STAR'];

var Index = 0;
var Flag = 0;

function Aendern()
{
    var AktuellerIndex = Index % 6 ; // MOD-Funktion

    ID_Div.filters[0].irisStyle = FeldDerIrisStyles[AktuellerIndex];
    ID_Span.innerText = 'IrisStyle = "' + FeldDerIrisStyles [AktuellerIndex] + '"';

    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="green";
    }

    ID_Div.filters[0].Play();

    Index += 1;
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Iris(duration=3);"
>
    Test-Text
<BR>
    Test-Text
<BR>

```



```

Test-Text
<BR>
</DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.irisStyle	Schärfe des Filters
.motion	Art der Bewegung
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.29. Light Filter

Lichtschein (Lichtquelle positionieren)

ab IE 5.5

Achtung: pro Webseite maximal 10 Lichtfilter als STYLE-Attribut kodierbar !!!
wenn mehr dann NUR Filterklasse definieren

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Light(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.Light(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<STYLE>
.aFilter { background-color: #FFFFFF; filter:light();color: cyan; width: 150;}
</STYLE>
<SCRIPT>
function Init()
{
    var iX2= ID_Div.offsetWidth;
    var iY2= ID_Div.offsetHeight;
    ID_Div.filters[0].addCone(0,0,1,iX2,iY2,255,0,0,20,180);
}
window.onload=Init;
</SCRIPT>
<DIV CLASS="aFilter" ID="ID_Div">
Test-Text
</DIV>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
----------	----------------------

Methoden:

.addAmbient()	indirektes Umgebungslicht (Ambiente)
.addCone()	direktes 3D-Licht als Lichttunnel oder Spot
.addPoint()	3D-Lichtpunkt als Lichtquelle, die in alle Richtungen strahlt
.changeColor()	Lichtfarbe nachträglich ändern
.changeStrength()	Lichtintensität nachträglich ändern
.clear()	alle Lichtquellen löschen
.moveLight()	Focus des Lichtkegels bzw. Lichtpunkt bewegen

4.3.2.2.4.2.2.5.30. MaskFilter Filter

transparente Pixelfarben maskieren so dass Umwandlung erfolgt von transparent zu nicht transparent UND nicht-transparente Pixelfarben zu transparent

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.MaskFilter(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.MaskFilter(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<STYLE>
DIV.aFilter { filter:mask(color=#008800); width: 100;}
</STYLE>
<DIV CLASS="aFilter">

```



Test-Text
</DIV>

Eigenschaften:

.color Farbe
.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.31. Matrix Filter

Filter zur rechenintensiven Drehung, Größenänderung, Rotation

nutzt Interpolation und lineare Transformation anhand einer Matrix mit Dimension 2 mal 2

	M11	M12	
	M21	M22	

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Matrix(Kette)" ... >
Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Matrix(Kette)"

Beispiel Kette String mit Eigenschaftenliste
Drehen und Ausdehnen (AutoExpand)

```
<SCRIPT>
    var Wert = Math.PI * (2 / 360); // Umrechnung für Winkelfunktionen

    function Rotation(DivZeiger, Faktor)
    {
        var Parameter = Faktor * Wert;
        var CosinusWert = Math.cos(Parameter);
        var SinuWert = Math.sin(Parameter);

        // Matrix belegen
        DivZeiger.filters.item(0).M11 = CosinusWert;
        DivZeiger.filters.item(0).M12 = -1 * SinuWert;
        DivZeiger.filters.item(0).M21 = SinuWert;
        DivZeiger.filters.item(0).M22 = CosinusWert;
    }

    function Ausdehnen(DivZeiger, Faktor)
    {
        // Matrix belegen
        DivZeiger.filters.item(0).M11 *= Faktor;
        DivZeiger.filters.item(0).M12 *= Faktor;
        DivZeiger.filters.item(0).M21 *= Faktor;
        DivZeiger.filters.item(0).M22 *= Faktor;
    }

    var Zahler = 400;

    function Start(DivZeiger)
    {
        var AusdehnFaktor = Zahler / 720;
        Zahler += 4;

        // wenn 3 volle Rotationen, so kein Filterereignis mehr verarbeiten
        if (Zahler >= 360*3)
        { DivZeiger.onfilterchange = null; }

        Rotation(DivZeiger, Zahler);
        Ausdehnen(DivZeiger, AusdehnFaktor);
    }
</SCRIPT>

<DIV STYLE="position:absolute;
    filter:progid:DXImageTransform.Microsoft.Matrix(sizingMethod='auto expand')"
    onfilterchange="Start(this)" >
    <DIV STYLE=" background-color: lightblue; padding:5;">
        Test-Text
    <BR>
        Test-Text
    <BR>
```




```

Test-Text
<BR>
Test-Text
</DIV>
</DIV>

```

Eigenschaften:

.Dx	X-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft . SizingMethod mit Wert "auto expand"
.Dy	Y-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft . SizingMethod mit Wert "auto expand"
.enabled	Filter-Anwendbarkeit
.FilterType	Filtertyp
.M11	Matrixfeld M11 (siehe oben)
.M12	Matrixfeld M12 (siehe oben)
.M21	Matrixfeld M21 (siehe oben)
.M22	Matrixfeld M22 (siehe oben)
.SizingMethod	Container-Resize ein/aus

Methoden:

keine

4.3.2.2.4.2.2.5.32. MotionBlur Filter

Bewegung im Objekt-Content

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.MotionBlur(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.MotionBlur(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<STYLE>
    DIV.aFilter { filter:progid:DXImageTransform.Microsoft.MotionBlur(Strength=5,Direction=90);}
</STYLE>
<DIV CLASS="aFilter" STYLE="width:200">
    Test-Text
</DIV>

```

Eigenschaften:

.add	Überlagerung des Filters über das Originalbild
.direction	Filterrichtung
.enabled	Filter-Anwendbarkeit
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.33. Pixelate Filter

farbiges Quadrat mit Pixeltransition und Deckkraftänderung

ab IE 5.5

benötigt Eigenschaft .enabled auf false

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Pixelate(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Pixelate(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.visibility="visible";
        }
        else
        {
            Flag = 1;
            ID_Div.style.visibility="hidden";
        }
    }

```



```

    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="width:100px; background-color: lightblue;
    filter:progid:DXImageTransform.Microsoft.Pixelate(duration=3, enabled='false');"
>
    Test-Text<BR>
    Test-Text<BR>
    Test-Text
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.maxSquare	Maximale Anzahl der Pixels im Quadrat
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.34. RadialWipe Filter

Scheibenwischer-Filter

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RadialWipe(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.RadialWipe(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<SCRIPT>
    var FeldDerWipeStyles = new Array();
    FeldDerWipeStyles = ['CLOCK', 'WEDGE', 'RADIAL'];

    var Zahler = 0;
    var Flag = 0;

    function Aendern()
    {
        var Index = Zahler % 3 ; // MOD Function

        ID_Div.filters[0].wipeStyle = FeldDerWipeStyles [Index];
        ID_Span.innerText = '.wipeStyle = "' + FeldDerWipeStyles [Index] + '"';

        ID_Div.filters[0].Apply();

        if (Flag)  // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="green";
        }

        ID_Div.filters[0].Play();
        Zahler += 1;
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;

```



```

        filter:progid:DXImageTransform.Microsoft.RadialWipe(duration=2);"
    >
        Test-Text
        <BR>
        Test-Text
        <BR>
        Test-Text
        <BR>
        Test-Text
    </DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.wipeStyle	Style des Wischens

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.35. RandomBars Filter

Zufallslinien

ab IE 5.5

Syntax:

```

HTML    <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RandomBars(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.RandomBars(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern ()
    {
        ID_Div.filters[0].orientation="vertical";
        ID_Div.filters[0].Apply();

        if (Flag)  // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.RandomBars(duration=5);"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.orientation	Orientierung
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück
----------	---



nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.5.36. RandomDissolve Filter

Zufallspixel

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RandomDissolve(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.RandomDissolve(Kette)"

Kette String mit Eigenschaftenliste
Beispiel

```
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.visibility="visible";
        }
        else
        {
            Flag = 1;
            ID_Div.style.visibility="hidden";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.RandomDissolve(duration=3);"
>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
</DIV>
```

Eigenschaften:

.duration Dauer der Animation des Filters
.enabled Filter-Anwendbarkeit
.Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status Status der Filteranimation

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.5.37. RevealTrans Filter

Sammlung von 24 Überblend-Filtereffekten

sichtbar wenn System-Farbpalette (System-Farbtiefe) mindestens 256 Farben

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:RevealTrans(Kette)" ... >
Scripting object.style.filter ="RevealTrans(Kette)"

Kette String mit Eigenschaftenliste

Beispiel



```

<SCRIPT>
    function go()
    {
        ID_Span.filters[0].Apply();
        if (ID_Span.style.visibility == "visible")
        {
            ID_Span.style.visibility = "hidden";
            ID_Span.filters.revealTrans.transition=2;
        }
        else
        {
            ID_Span.style.visibility = "visible";
            ID_Span.filters[0].transition=3;
        }
        ID_Span.filters[0].Play();
    }
</SCRIPT>
<INPUT TYPE="BUTTON" VALUE="Play Transistion" onclick="go();">
<SPAN ID="ID_Span" STYLE="position:absolute; Visibility:visible;
    Filter:revealTrans(duration=2, transition=3);
    width:300; height:300; background-color: lightgreen;"
>
    <CENTER style="background-color=red; height=100; width:100; position:relative; top:100">
    </CENTER>
</SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.transition	Filternummer

numerischer Wert der Eigenschaft transition**alternative Kodierung ab IE 5.5**

0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.38. Shadow Filter

Konturschatten

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Shadow(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Shadow(Kette)"



Kette String mit Eigenschaftensliste

Beispiel für Schatten eines DIV:

```
<STYLE>
    DIV.aFilter { filter:shadow(color=#0000FF,direction=45); width: 150; color: FF0000;}
</STYLE>
<DIV CLASS="aFilter">
    Test-Text
</DIV>
```

Beispiel für Schatten eines Bildes und eines Textes:

```
<HTML>
<HEAD>
<STYLE>
    DIV.filter_name { filter:shadow(color=#000000,direction=120); width:100; color: #FF0000;}
</STYLE>
</HEAD>
<BODY>
    <DIV CLASS="filter_name" STYLE="width:400;">
        <IMG SRC="bild.gif">
        Das Bild und dieser Text haben einen Schatten
    <BR>
    Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
    </DIV>
</BODY>
</HTML>
```

Beispiel für 3D-Effekt eines Bildes:

Dieser Effekt dient dem verschobenen Überlagern ein und desselben Bilder, so dass ein 3D-Effekt mit Schatten erzeugt wird. Die Verschiebung ist damit die Schattendicke bzw. die Effekt-Stärke des 3D-Effektes. Besonders günstig sieht man diesen Effekt, wenn die Grafik einen Schriftzug enthält.

Die Überlagerung erfolgt anhand zweier DIV's, die je eine CSS-Klasse erhalten und ansonsten identisch sind.

```
<HTML>
<HEAD>
<STYLE>
    DIV.DIV1_filter:{ filter:shadow(color=#FFFF00,direction=300, strength=5); width:100; color: #FF0000;}
    DIV.DIV2_filter:{ filter:shadow(color=#000000,direction=120, strength=8); width:100; color: #FF0000;}
</STYLE>
</HEAD>
<BODY>
    <DIV CLASS="DIV1_filter " STYLE="width:400; position:absolute; top:50px; left:50px;">
        <BLOCKQUOTE>
            <IMG SRC="bild.gif">
            Das Bild und dieser Text haben einen Schatten
            <BR>
            Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
        </BLOCKQUOTE>
    </DIV>
    <DIV CLASS="DIV2_filter " STYLE="width:400; position:absolute; top:50px; left:50px;">
        <BLOCKQUOTE>
            <IMG SRC="bild.gif">
            Das Bild und dieser Text haben einen Schatten
            <BR>
            Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
        </BLOCKQUOTE>
    </DIV>
</BODY>
</HTML>
```

Eigenschaften:

.color	Farbe
.direction	Filterrichtung
.enabled	Filter-Anwendbarkeit
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.39. Slide Filter

ab IE 5.5



Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Slide(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Slide(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var FeldDerSlideTyles = new Array();
FeldDerSlideTyles = ['HIDE', 'PUSH', 'SWAP'];

var Zahler = 0;
var Flag = 0;

function Aendern()
{
    var Index = Zahler % 3 ; // MOD-Funktion
    ID_Div.filters[0].slideStyle = FeldDerSlideTyles[Index];
    ID_Span.innerText = 'slideStyle = ' + FeldDerSlideTyles[Index] + '";

    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="green";
    }

    ID_Div.filters[0].Play();

    Zahler += 1;
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Slide(duration=3, bands=8);"
>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
</DIV>
<SPAN ID="ID_Span" oSpan"></SPAN>
```

Eigenschaften:

.bands	Anzahl der Streifen
.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.slideStyle	Art
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.40. Spiral Filter

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Spiral(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Spiral(Kette)"



Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
var Flag = 0;
function Aendern()
{
    ID_Div.filters[0].Apply();
    if (Flag)
    {
        Flag = 0;
        ID_Div.style.visibility="visible";
    }
    else
    {
        Flag = 1;
        ID_Div.style.visibility="hidden";
    }
    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Spiral(
duration=3, GridSizeX=25, GridSizeY=25);"
>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.gridSizeX	Anzahl Spalten des Gitters
.gridSizeY	Anzahl Zeilen des Gitters
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.41. Stretch Filter

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Stretch(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Stretch(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
var FeldDerStretchStyles = new Array();
FeldDerStretchStyles = ['HIDE', 'PUSH', 'SPIN'];

var Zahler = 0;
var Flag = 0;

function Aendern()
{
    var Index = Zahler % 3 ; // MOD-Funktion
    ID_Div.filters[0].stretchstyle = FeldDerStretchStyles[Index];
}

```




```

        ID_Span.innerText = '.stretchStyle = "' + FeldDerStretchStyles[Index] + '"';

        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="green";
        }

        ID_Div.filters[0].Play();

        Zahler += 1;
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
        filter:progid:DXImageTransform.Microsoft.Stretch(duration=3);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
</DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.stretchStyle	Art

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.42. Strips Filter

Diagonalstreifen

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Strips(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Strips(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {

```



```

        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Strips(
        duration=5, motion='rightdown');"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.motion	Ecke
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.stretchStyle	Art

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.43. Wave Filter

vertikale Wellen

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Wave(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Wave(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<STYLE>
    DIV.aFilter { filter:wave(strength=2, freq=3, lightstrength=20, add=0, phase=90);
        width: 150; color: #FF0000;}
</STYLE>
<DIV CLASS="aFilter">
    Test-Text
</DIV>

```

Eigenschaften:

.add	Überlagerung des Filters über das Originalbild
.enabled	Filter-Anwendbarkeit
.freq	Anzahl der Wellen
.lightStrength	prozentuale Differenz der Licht-Intensität zwischen Wellengipfel und Wellentäler
.phase	prozentuale Phasenverschiebung der Wellen, also Verschiebung der Wellenüberlagerung als Prozentanteil des Wellenzyklus
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.44. Wheel Filter

Speichenrad-Dreh-Filter

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Wheel(Kette)" ... >
Scripting	object.style.filter = "progid:DXImageTransform.Microsoft.Wheel(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {

```



```

        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
        filter:progid:DXImageTransform.Microsoft.Wheel(duration=2, spokes=8);"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.spokes	Anzahl Speichen
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.45. Xray Filter

Schwarz-Weiss Darstellung

ab IE 4.x

Syntax:

HTML	<ELEMENT STYLE="filter:Xray" ... >
Scripting	object.style.filter ="Xray"

Beispiel

```

<STYLE>
    DIV.aFilter { filter:xray; width: 150; color: #FF0000;}
</STYLE>
<DIV CLASS="aFilter">
    Test-Text
</DIV>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
----------	----------------------

Methoden:

keine

4.3.2.2.4.2.2.5.46. ZigZag Filter

Zigzag-Bewegung

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Zigzag(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.Zigzag(Kette)"

Kette	String mit Eigenschaftenliste
-------	-------------------------------

Beispiel

```

<SCRIPT>
    var Flag = 0;
    function Aendern()
    {
        ID_Div.filters[0].Apply();
        if (Flag)
        {

```



```

        Flag = 0;
        ID_Div.style.visibility="visible";
    }
    else
    {
        Flag = 1;
        ID_Div.style.visibility="hidden";
    }
    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Zigzag(
    duration=3, GridSizeX=25, GridSizeY=25);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.gridSizeX	Anzahl Spalten des Gitters
.gridSizeY	Anzahl Zeilen des Gitters
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.47. Filter bei wichtigen Objekten (Auswahl)**a Objekt**

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

area Objekt

keine Filter

applet Objekt

keine Filter

body Objekt

Alpha	CheckerBoard	FlipV
AlphaImageLoader	Chroma	Glow
Barn	Compositor	Gradient
BasicImage	Emboss	GradientWipe
BlendTrans	Engrave	Gray
Blinds	Fade	ICMFilter
Blur	FlipH	Inset



Invert	RadialWipe	Stretch
Iris	RandomBars	Strips
Light	RandomDissolve	Wave
MaskFilter	RevealTrans	Wheel
Matrix	Shadow	Xray
MotionBlur	Slide	Zigzag
Pixelate	Spiral	

button Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

currentStyle

keine

custom Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

div Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

document Objekt

keine Filter

fieldset Objekt

Alpha	Chroma	Glow
AlphaImageLoader	Compositor	Gradient
Barn	DropShadow	GradientWipe
BasicImage	Emboss	Gray
BlendTrans	Engrave	ICMFilter
Blinds	Fade	Inset
Blur	FlipH	Invert
CheckerBoard	FlipV	Iris



Light
MaskFilter
Matrix
MotionBlur
Pixelate
RadialWipe

RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral

Stretch
Strips
Wave
Wheel
Xray
Zigzag

font Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave Performs
Wheel
Xray
Zigzag

form Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

frame Objekt

Alpha
AlphaImageLoader
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Engrave
Fade
FlipH

FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur
Pixelate

RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Wave
Wheel
Xray
Zigzag

frameset Objekt

Barn
BasicImage
Blinds
Blur
CheckerBoard
Compositor
Emboss
Engrave
Fade

Gradient
GradientWipe
ICMFilter
Inset
Iris
Matrix
Pixelate
RadialWipe
RandomBars

RandomDissolve
Slide
Spiral
Stretch
Strips
Wheel
Zigzag

iframe Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans

Blinds
Blur
CheckerBoard
Chroma
Compositor

DropShadow
Emboss
Engrave
Fade
FlipH



FlipV	Light	Shadow
Glow	MaskFilter	Slide
Gradient	Matrix	Spiral
GradientWipe	MotionBlur	Stretch
Gray	Pixelate	Strips
ICMFilter	RadialWipe	Wave
Inset	RandomBars	Wheel
Invert	RandomDissolve	Xray
Iris	RevealTrans	Zigzag

img Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	ZigZag

input Objekt

keine Events

input button Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input checkbox Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input file Objekt

Alpha	Compositor	GradientWipe
AlphaImageLoader	DropShadow	Gray
Barn	Emboss	ICMFilter
BasicImage	Engrave	Inset
BlendTrans	Fade	Invert
Blinds	FlipH	Iris
Blur	FlipV	Light
CheckerBoard	Glow	MaskFilter
Chroma	Gradient	Matrix



MotionBlur
Pixelate
RadialWipe
RandomBars
RandomDissolve

RevealTrans
Shadow
Slide
Spiral
Stretch

Strips
Wave
Wheel
Xray
Zigzag

input hidden Objekt

keine Filter

input image Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

input password Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

input radio Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

input reset Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave

Fade
FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter

Matrix
MotionBlur
Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave



Wheel

Xray

Zigzag

input submit Objekt

Alpha

FlipH

Pixelate

AlphaImageLoader

FlipV

RadialWipe

Barn

Glow

RandomBars

BasicImage

Gradient

RandomDissolve

BlendTrans

GradientWipe

RevealTrans

Blinds

Gray

Shadow

Blur

ICMFilter

Slide

CheckerBoard

Inset

Spiral

Chroma

Invert

Stretch

Compositor

Iris

Strips

DropShadow

Light

Wave

Emboss

MaskFilter

Wheel

Engrave

Matrix

Xray

Fade

MotionBlur

Zigzag

input text Objekt

Alpha

FlipH

Pixelate

AlphaImageLoader

FlipV

RadialWipe

Barn

Glow

RandomBars

BasicImage

Gradient

RandomDissolve

BlendTrans

GradientWipe

RevealTrans

Blinds

Gray

Shadow

Blur

ICMFilter

Slide

CheckerBoard

Inset

Spiral

Chroma

Invert

Stretch

Compositor

Iris

Strips

DropShadow

Light

Wave

Emboss

MaskFilter

Wheel

Engrave

Matrix

Xray

Fade

MotionBlur

Zigzag

marquee Objekt

Alpha

FlipH

Pixelate

AlphaImageLoader

FlipV

RadialWipe

Barn

Glow

RandomBars

BasicImage

Gradient

RandomDissolve

BlendTrans

GradientWipe

RevealTrans

Blinds

Gray

Shadow

Blur

ICMFilter

Slide

CheckerBoard

Inset

Spiral

Chroma

Invert

Stretch

Compositor

Iris

Strips

DropShadow

Light

Wave

Emboss

MaskFilter

Wheel

Engrave

Matrix

Xray

Fade

MotionBlur

Zigzag

object Objekt

Alpha

FlipH

Pixelate

AlphaImageLoader

FlipV

RadialWipe

Barn

Glow

RandomBars

BasicImage

Gradient

RandomDissolve

BlendTrans

GradientWipe

RevealTrans

Blinds

Gray

Shadow

Blur

ICMFilter

Slide

CheckerBoard

Inset

Spiral

Chroma

Invert

Stretch

Compositor

Iris

Strips

DropShadow

Light

Wave

Emboss

MaskFilter

Wheel

Engrave

Matrix

Xray

Fade

MotionBlur

Zigzag

option Objekt

keine

runtimeStyle

Alpha

BlendTrans

Chroma



DropShadow
FlipH
FlipV
Glow
Gray

Invert
Light
MaskFilter
MotionBlur
Redirect

RevealTrans
Shadow
Wave
Xray

select Objekt

keine Filters

selection Objekt

keine Filters

span Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

style Objekt

Alpha
BlendTrans
Chroma
DropShadow
FlipH
FlipV

Glow
Gray
Invert
Light
MaskFilter
MotionBlur

RevealTrans
Shadow
Wave
Xray

table Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.caption Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.col Objekt

keine



table.colGroup Objekt

keine

table.tBody Objekt

keine

table.tFoot Objekt

keine

table.tHead Objekt

keine

table.tr Objekt

keine

table.tr.td Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade
FlipH

FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur
Pixelate
RadialWipe

RandomBars
RandomDissolve
Redirect
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.tr.th Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade
FlipH

FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur
Pixelate
RadialWipe

RandomBars
RandomDissolve
Redirect
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

textarea Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

var Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans

Blinds
Blur
CheckerBoard
Chroma
Compositor

DropShadow
Emboss
Engrave
Fade
FlipH



FlipV	Light	Shadow
Glow	MaskFilter	Slide
Gradient	Matrix	Spiral
GradientWipe	MotionBlur	Stretch
Gray	Pixelate	Strips
ICMFilter	RadialWipe	Wave
Inset	RandomBars	Wheel
Invert	RandomDissolve	Xray
Iris	RevealTrans	Zigzag

window Objekt

keine Filter

4.3.2.2.4.3. HTML-Elemente im HTML-Dokument des Internet Explorer (Auswahl)**Voraussetzungen für die programmtechnische Nutzung von HTML-Elementen als Objekte in JScript:**

Kenntnisse der Programmierers zu:

DOM
 Zeigerverwendung
 Vererbung
 Browserversionen/Script-Maschinen-Versionen
 JScript-Sprache
 HTML-Syntax zum HTML-Element (Attribute etc.)

Hinweise: JScript von Microsoft ist programmtechnisch nicht netzwerkfähig, es sei denn, es werden Komponenten des Betriebssystems
 Windows

in JScript aktiviert (falls überhaupt möglich). JScript ist nur client-orientiert und arbeitet direkt auf dem PC des Internet-Users, der eine Webseite mit JScript-Teilen besucht. Aus Sicht des Webseitenbetreibers (Hoster) und dessen Server ist der User der "Endverbraucher" des Internets, also der Klient ("Kunde") des Webseitenangebotes. Wer netzwerkfähige Produkte erzeugen will, sollte sich mit der NET-Software zu Windows XP von Microsoft beschäftigen, die JScript und XML auf Basis einer **installierten und aktiven** Server-Software netzwerkfähig machen.

Bezüglich des implementierten HTML-DOM vom Internet Explorer ist festzustellen, dass große Ähnlichkeiten bzw. teilweise Identität mit anderen DOM, z.B. von XML, vorliegen. Für den Programmierer, der nicht nur JScript nutzen will, ist es für den Einstieg in XML vorteilhaft, die Prinzipien des HTML-DOM und dessen Umsetzung in JScript verinnerlicht zu haben. Microsoft setzt ganz bewusst auf prinzipielle Gemeinsamkeiten und Integration der Script-Programmierung per JScript mit bzw. in XML. So gesehen wird XML das HTML problemlos ablösen.

JScript kann nur dann auf dem User-PC aktiv werden, wenn der User Javascript zugelassen hat. Die Zulassung von Javascript hängt von der Browser-Software und optional eingesetzter Firewall-Software ab.

JScript kann Komponenten des Betriebssystems Windows wie DirectX oder ActiveX nutzen. Auch hier gilt: Der User muss die Nutzung von DirectX und ActiveX zugelassen haben. Es gibt Webseitenanbieter, z.B. Hoster für kostenlose Homepages, deren Host-Angebote (Webseiten der Kunden des Hosters) **nur** unter Nutzung von ActiveX funktionieren, wenn die Homepage mit Komponenten des Hosters erstellt wurden. Dem Besucher solcher Webseiten ist in der Regel nicht bekannt, was die ActiveX-Komponenten bewirken. Solche Webseitenangebote sind dann aus Sicht des Schutzes der Privatsphäre des Users **wertlos**, wenn der User ActiveX gesperrt hat (weil kein Risiko eingehen will) und in der Webseite keine Alternativen zu ActiveX bzw. Hinweise zu Art und Zweck von eingesetzten ActiveX-Komponenten programmiert wurden.

Der JScript-Programmierer muss daher **regelmäßig** mit dem Einsatz von Firewall-Software auf dem User-PC rechnen und daher den Einsatz von Windowskomponenten für den User **transparent** halten, wenn der Programmierer möchte, dass **sicherheitsbewusste** User die erstellte Webseite nutzen können und der User **keine Risiko** tragen soll.

JScript ist im Gegensatz zu JavaScript von Netscape nur schwer mit JAVA verbindbar, da die Browserhersteller divergierende Interessen zu JAVA besitzen und somit Bezüge auf JAVA mehr oder weniger in die Scriptmaschine implementieren.

JScript ist wie jede JavaScript-Sprache eine rein objektorientierte Sprache, die mit Zeigern arbeitet., denn HTML-Elemente können **nur als Objekte**, also nur objektorientiert verarbeitet werden. Die Zeigernutzung erfolgt intern, aber im Regelfall auch durch den Programmierer per Script-Code, vorallem dann sehr intensiv, wenn der Programmierer einen dynamischen Inhalt der Webseite erstellen will.

Die Pflege und Dokumentation der Scriptsprache und deren Implementationen im Browser hängt vom Willen des Herstellers ab, der mehr oder weniger in der Lage sein möchte, für denjenigen Transparenz zu gewähren, der Script programmtechnisch nutzen will. Damit können Objektbeschreibungen kurz- oder langzeitig sein, letzteres nur dann, wenn Browser-Versionen des Herstellers kompatibel sind bzw. Unterschiede regelmäßig qualifiziert und leicht zugänglich publiziert werden.

Abbildung des HTML-Elementes als Objekt:

HTML-Elemente werden als Objekte im Rahmen des HTML-DOM erzeugt.

Attribute des HTML-Elementes laut HTML-Syntax sind im Objekt als Eigenschaften abgebildet, leider - je nach Art des HTML-Elementes - **nicht komplett (nicht alle Attribute)**. Und: Es existieren z.T. Eigenschaften, denen der jeweilige Pedant in HTML-Attribut-Form fehlt - besonders beim Internet Explorer, der damit wesentlich mehr als HTML bietet, das allerdings Microsoft-spezifisch für das Betriebssystem Windows. Diese Abbildungs-Varianten sind der Ausdruck des Grades einer HTML-Unterstützung im Browser, denn HTML-Elemente können **nur als Objekte**, also nur objektorientiert verarbeitet werden. JScript ist daher, wie jede JavaScript-Sprache, eine rein objektorientierte Sprache, die grundsätzlich mit Zeigern auf Instanzen eines Objektes arbeitet.



Methoden des Objektes, das das HTML-Element abbildet,

dienen nur z.T. dem Aktions-Zweck des Elementes im HTML-Design. Beispiel MARQUEE-Element (scrollender Text):

Während in HTML das Element scheinbar selbständig mit vorher fest definierten Attribut-Werten startet, ist **dasselbe** Element als Objekt in JScript nach dessen Start dynamisch steuerbar - u.a. auch anhand der Methoden zum Objekt. (Selbst Werte von in HTML fest definierten Attributen sind per Objekt-Eigenschaften dynamisch zur Aktionszeit des HTML-Elementes (zur Laufzeit des Objektes im Dokument) abänderbar.),

lassen z.B. auch steuerbare Kommunikation zwischen anderen HTML-Elementen per Ereignisse (Events) zu, ermöglichen z.B. Manipulation von Vererbungs-Hierarchien, können z.B. Kind-Objekte erzeugen und das Layout des HTML-Elementes verändern.

Das HTML-Objekt wird nicht 1:1 abgebildet. Der Programmierer kann aber den Zusammenhang zwischen HTML-Element und Objekt erkennen (z.B. häufige Bezeichneridentität von Attribut im HTML-Element und Eigenschaft im abbildenden Objekt).

JScript kann HTML-Code verwenden und erzeugen, muss es aber je nach Wunsch des Programmierers nicht: Wenn das HTML-DOM es zulässt, sind HTML-Elemente per JScript ohne irgend einen HTML-Code im JScript-Code komplett erzeugbar.

Per Methoden des Objektes `window.document.write()` oder `window.document.writeln()` programmtechnisch erzeugte HTML-Elemente werden browserintern immer mit den Methoden des DOM erzeugt. Alternativ zu `write()` und `writeln()` kann der Programmierer auch gleich die DOM-Methoden in JScript kodieren.

Zeigerbezüge (Objekt-Referenzen) auf HTML-Elemente: ***im Dokument***

HTML-Elemente sind Objekte des Dokumentes, also Kinder des Objektes `document`. Die Referenzierung der HTML-Elemente im Dokument erfolgt in der Regel per

`Collection document.all`
oder per Zeiger laut ID-Attribut bzw. NAME-Attribut.
oder per spezieller Collectionen (je nach HTML-Elemente-Art bzw. DOM)
oder per DOM (Document Object Model) des HTML

Collectionen sind Felder aus Zeigern auf Instanzen je nach Art der Collection (siehe Beschreibungen der Collectionen). existieren z.T. nur spezifisch zu einer HTML-Elemente-Art oder sind Komponenten des DOM (Document Object Model) sind immer in der Scriptmaschine zum Browser implementiert (kurz: "im Browser implementiert") haben immer genau definierte Eigenschaften und Methoden.

Hinweis: Programmtechnisches Analogon ist das Feld (array Objekt).

Alle Zeigerbezüge nutzen ein DOM, das im Browser (je nach Version der Scriptmaschine, also je nach Browser-Version) fest implementiert ist. Hinweis: Es gibt mehrere DOM. Nachfolgende Betrachtungen nutzen das DOM zu HTML und z.T. zu XML.

im Fenster

Da das Dokument in einem Fenster eingebettet ist, sind HTML-Elemente auch Kinder des dem Objekt `document` zugehörigen Eltern-Objektes `window`.

im aktuellen Fenster und dessen Dokument (kurz "im aktuellen Dokument")

Der komplette Objektpfad lautet

`window.document.all.object_zeiger.eigenschaft`
`window.document.all.object_zeiger.methode()`

`window.objekt_zeiger_laut_id_attribut_bzw_name_attribut.eigenschaft`
`window.objekt_zeiger_laut_id_attribut_bzw_name_attribut.methode()`

Achtung: Die Pfad-Komponente `window.document.` ohne `.all` wird nur dann ohne Fehler vom Interpreter akzeptiert, wenn nicht explizit die Collection `document.all` benötigt wird.

Zur Vereinheitlichung der Zeigerbezüge ist die Verwendung des Zeigers laut im HTML-Element kodiertem ID-Attribut wärmstens zu empfehlen, da dieser Zeiger immer im korrekten Kontext erzeugt wird und die Pfadkomponente `window.document.all` nicht zusätzlich kodiert werden muss, solange kein Bezug auf `document.all` explizit nötig ist (nur bestimmte Objekte oder Collectionen müssen per `document.all` referenziert werden).

Üblicherweise wird der Suffix (Pfadkomponente) `window.`
oder `window.document.all.`
nicht kodiert, wenn es sich um das Dokument im aktuellen Fenster handelt (ansonsten ist der Zeiger auf das konkrete Fenster und dessen Dokument zu kodieren). Dabei ist die Browser-Performance zu beachten (siehe tiefer).

Achtung: Zur Browserunterscheidung ist es wichtig, dass der Programmierer weiss, welche Kind-Objekte vom Objekt `document` unterstützt werden.

Es ist Fakt, dass neben dem Internet Explorer auch andere Browser Objekte zum Objekt `document` implementiert haben. Damit ist die Referenz `document.objekt_zeiger`



nicht **nur** für den Internet Explorer gültig.

Soll aber nur der Internet Explorer beachtet werden, dann ist document.all zu kodieren, vorallem dann, wenn ein Kind-Objekt des Objektes document in mehreren Browsern implementiert ist, aber dort mit **verschiedenen** Eigenschaften oder Methoden.

Damit sind IE-spezifische Eigenschaften zum Objekt per document.all.objekt_zeiger.eigenschaft zu kodieren (analog zu Methoden).

Nachfolgende Objekt- und Collectionen-Beschreibungen

lassen die Referenz per window.document.all

oder document.all

weg und gehen von der Nutzung des ID-Attributes aus, dessen Wert als Objekt-Zeiger dient verwenden im Bezeichner des Objektes nur dann den Suffix document., wenn deutlich gemacht werden soll, dass das Objekt **nur ein Kind vom Objekt document sein kann**, also nicht als Kind eines anderen Objektes erzeugt werden darf. Wichtig ist dieser Umstand auch bei einigen Collectionen.

Bsp. document.body Objekt Body kann nur Kind von document sein

Bsp. a Objekt a kann Kind eines anderen Kindes von document sein.

verzichten auf eine Baumhierarchie der Objekte, da wie in HTML mehr **frei gestaltbare** Verschachtelungsvarianten zulässig sind, als im DOM fest vorgeschriebene. Im Prinzip gilt: Jedes HTML-Dokument hat als Ergebnis des Webdesign **seine eigene, spezifische Hierarchie**.

stellen Objekte und Collectionen als Systematik in alphabetischer Reihenfolge bzw. im Sinnzusammenhang dar. Es gibt natürlich auch andere Gesichtspunkte der Objektbeschreibungen, die bereits weiter oben in dieser Dokumentation dargestellt wurden, z.B. Objekte und Collectionen zur Verwaltung aller bzw. spezieller HTML-Elemente oder die Beschreibung des DOM. Die Trennung dieser Beschreibungen von denen der nachfolgend beschriebenen Objekte bewirkt, dass z.T. Objekte nicht nachfolgend, sondern an anderer Stelle beschrieben werden bzw. wurden. Das bedingt sich aufgrund der Systematik dieser Dokumentation. Als Empfehlung zum Nachschlagen wird zusätzlich das Inhaltsverzeichnis bzw. der Index nahegelegt.

sind in ihrer Objekt-Hierarchie am besten aufzuarbeiten, wenn mit dem Lesen der Beschreibung zum Body-Objekt begonnen wird. Ein weiteres und sehr oft in der JScript-Programmierung verwendetes Objekt ist das div Objekt. Beide genannten Objekte sind wichtige Container für HTML-Elemente innerhalb des Dokumentes, also wichtige Kinder des Objektes document.

zum Layout (Style) vom Dokument und dessen HTML-Elemente liegen bezüglich **aller** zugehörigen Objekte und Collectionen **nur** innerhalb der Beschreibung des **style Objektes** (STYLE-Elementes) und **nicht in alphabetischer Folge**. Style ist aufgrund seiner Komplexität und als weiteres wichtiges Objekt im Webdesign nur als Einheit beschreibbar.

zu den zusätzlichen Standard-Verhaltensweisen (Standard-Behavior) von HTML-Elementen liegen als Komponenten des style Objektes bezüglich **aller** zugehörigen Objekte und Collectionen zu den Verhaltensweisen **nur** innerhalb der Beschreibung des **style Objektes** (STYLE-Elementes) und **nicht in alphabetischer Folge**. Style ist aufgrund seiner Komplexität und als weiteres wichtiges Objekt im Webdesign nur als Einheit beschreibbar.

Hinweis: Aus Gründen der Browser-Performance sind:

sämtliche Objekte, die das ID- bzw. NAME-Attribut unterstützen, mit diesem Attribut zu versehen, wenn mindestens 1 im JScript-Quellcode kodierter Zeiger-Bezug auf das abbildende Objekt existiert: Der Wert des Attributes ist der Zeigerbezeichner auf das Objekt. (Aus Gewohnheitsgründen einfach immer das ID-Attribut kodieren z.B. für im Design sich erst später als notwendig erweisende Bezüge auf das Objekt). Wenn ID **und** NAME unterstützt werden, so ID kodieren, da NAME nur im Spezialfall kodiert werden muss, z.B. im Formular-Element.

Bsp. ohne ID-Attribut-Kodierung:

```
document.write('<ELEMENT></ELEMENT>');
document.all[index_auf_element].style.visibility='hidden';

// der Index auf die Collection document.all muss
// dem Programmierer bekannt sein und direkt in den Quellcode
// kodiert werden
// oder vorher als Variable programmtechnisch ermittelt worden sein
// var index_auf_element= .....;
// ELEMENT ist Platzhalter für ein konkretes HTML-Element-Tag
```

Bsp.: mit ID-Attribut-Kodierung:

```
document.write('<ELEMENT ID="ZeigerAufDasElement"></ELEMENT>');
ZeigerAufDasElement.style.visibility='hidden'; // keine var-Anweisung !!!

// der Index auf die Collection document.all ist automatisch ermittelt, so dass
// die Kodierung von document.all entfallen kann
// ELEMENT ist Platzhalter für ein konkretes HTML-Element-Tag
```

Punktnotationen, die gemeinsame Komponenten besitzen, sich aber ansonsten unterscheiden, zu zerlegen: Die gemeinsamen Komponenten sind aufzulösen und je als 1 eigenständiger Zeiger zwischenspeichern.

Bsp.: window_zeiger.document_zeiger.object_zeiger



zerlegen in

```

var ZeigerAufAktuellesDokument = window_zeiger.document_zeiger;
var ZeigerAufObjektImAktuellenDokument =
    ZeigerAufAktuellesDokument.object_zeiger;
.....
var ZeigerAufObjektImAktuellenDokument_AufStyleEigenschaft=
    ZeigerAufObjektImAktuellenDokument.style; // 1x berechnet

ZeigerAufObjektImAktuellenDokument_AufStyleEigenschaft.visibility='hidden';
ZeigerAufObjektImAktuellenDokument_AufStyleEigenschaft.cursor='hand';
....
ZeigerAufObjektImAktuellenDokument.stop();
.....
ZeigerAufObjektImAktuellenDokument=null;

```

Ziele der Zerlegung, also der Auflösung der Punktnotation auf einen Minimalumfang, sind:
 Vermeidung von Mehrfachkodierungen von identischen Punktnotationen im Quellcode
 Vermeidung der pro erkannten und unaufgelösten Punktnotation durchgeführten
 Referenzberechnung (Zeigerberechnung), so dass nur für den zwischengespeicherten
 Zeiger die Referenz genau 1x berechnet werden muss und danach der bereits
berechnete
 Zeiger mehrfach verwendet wird.

Hinweis zur Referenzierung von Eigenschaften und Methoden:

Diverse Objekte und Collectionen haben bezeichner-identische Eigenschaften und Methoden. Daher sind die exakte Referenzierung der Objekte und Collection und das Wissen, welche Eigenschaften und Methoden zu den Objekten und Collectionen implementiert sind, wichtig.

4.3.2.2.4.3.1. a Objekt des Internet Explorer

Hypertext-Link (Linkt mit Textangabe für User) also Anker:

Es kann auch ein Image im Text angegeben werden (zusätzlich oder anstelle des Textes).

Bei Image: Border unsichtbar durch Eigenschaft .border auf 0 setzen.

Achtung: keine Tabelle einbindbar (kein Objekt table) !!

Script einbindbar

benötigt **immer** die kodierte Eigenschaft .href oder .name

nicht verschachtelbar

Objekt mit Timeline nur wirksam, wenn diese aktiv ist.

siehe auch link Objekt

Erzeugung:

Beispiele:

```

<A ID="ID_A" HREF="http://www.test.de.com">gehe zu test.de</A>

<A ID="ID_A" HREF="home.htm">home.htm</A>

<A ID="ID_A" TARGET="name_des_fensters" HREF="test.htm">im Fenster oeffnen</A>

<A ID="ID_A" HREF="http://www.test.de "><IMG SRC="test.gif">Link mit Image und Text</A>

<A ID="ID_A" NAME="anchor_name">Eine Anker als Sprungziel</A>

<A ID="ID_A" HREF="#anchor_name">Anker ausführen </A>

<A ID="ID_A" HREF="javascript:window.open()">Javascript</A>

<A HREF="text.doc">Den Text als Word-Dokument herunterladen</A>

<HEAD>
<STYLE>
    .an {text-decoration: underline overline; color:blue;}
    .aus {text-decoration: none; color:black;}
</STYLE>
</HEAD>
<BODY>
    <A
        HREF="test.htm"
        CLASS="aus"
        onmouseover="this.className='an';"
        onmouseout="this.className='aus';"
    >
    </A>
</BODY>

```

Zugriff:

```

ID_A.eigenschaft
ID_A.methode()

```



Beispiel 1:

```
<HTML XMLNS:IE>
<HEAD>
<SCRIPT>
    function NachDemDownload(DownLoadedFileContent)
    {alert (DownLoadedFileContent);}
</SCRIPT>
</HEAD>
<BODY>
<IE:Download ID="ID_IETag" STYLE="behavior:url(#default#download)" >
<P>
Click
<A HREF="javascript:ID_IETag.startDownload('download.htm', NachDemDownload)">
hier
</A>
zum Start des Downloads dieser Seite.
</BODY>
</HTML>
```

Beispiel 2 für Link ohne Linkrahmen bei IE
NS ab 6.x

Achtung: Eventuell wird Navigation per TAB-Taste nicht mehr möglich sein !

Dies ist ein Link ohne Rahmen bei Focuserhalt

Dies ist ein Link ohne Rahmen nach klick

Beispiel 3 für Link ohne Unterstreichung:

```
<HTML>
<HEAD>
    <STYLE TYPE="text/css">
        A { text-decoration: none; }
    </STYLE>
</HEAD>

<BODY>
    <A HREF="http://www.test.de/">Ein Link ohne Unterstreichung</A>
</BODY>
</HTML>
```

Alternative:

Beispiel:

```
<FORM ACTION="test.htm">
    <INPUT TYPE="submit" VALUE="Gehe zu test.htm">
</FORM>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird fokussiert die Sprungquelle defokussiert das Focus-Ereignis ausgelöst
ATOMICSELECTION	vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.charset	Zeichensatz zum Encoden eines Objektes im Dokument
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.coords	Koordinaten eines Objektes in Abhängigkeit zur Eigenschaft .shape
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen



.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hash	Teil des Wertes der Eigenschaft .href also Teil der Url als Anker Teil folgt direkt hinter dem Nummernkreuz # und ohne Nummernkreuz
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Fokussierbarkeit
.host	Hostname und Port einer Location oder Url in der Form " hostname:port "
.hostname	Hostname einer Location oder Url in der Form " hostname:port " kann Domain oder IP sein
.href	Ziel-Url oder Anker als Sprungziel (z.B. <A>-Tag) siehe auch Eigenschaften .rel und .rev
.hreflang	Sprachcode des Objektes laut RFC1766
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.Methods	Liste der vom Objekt unterstützten HTTP-Methoden
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes Collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie



.parentTextEdit	Textbereich des Elternobjektes referenzieren
.pathname	Datei und Pfad eines Objektes
.port	Port einer Location oder Url in der Form "hostname: port " basierend auf dem aktuellen Protokoll laut Eigenschaft .protocol z.B. <u>Standard-Ports</u> <u>Protokoll</u> 21 FTP 80 HTTP
.previousSibling	Referenz auf das Vorgängerkind
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.rel	Beziehung zwischen Objekt in Quellseite und Nachfolger-Zielseite laut Objekt Objekt ist ein Link z.B. A-Tag oder Link-Tag nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen beim durchforsten der Seiten und Nachfolgerseiten vorrangig kodiert in <A> oder <LINK> es muss zugleich die Eigenschaft .href kodiert und mit gültigen Wert belegt sein nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)
.rev	Beziehung zwischen Objekt in Quellseite und Vorgänger-Zielseite laut Objekt Objekt ist ein Link z.B. A-Tag oder Link-Tag nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen beim durchforsten der Seiten und Vorgängerseiten vorrangig kodiert in <A> oder <LINK> es muss zugleich die Eigenschaft .href kodiert und mit gültigen Wert belegt sein nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.search	Teil des Wertes des Eigenschaft .href Teil folgt direkt dem Fragezeichen wird als Querystring oder Formdata bezeichnet hat nichts mit der Suche auf Webseiten zu tun Hinweis: Fragezeichen-Anhang an der HREF dient zur Übergabe von String-Werten einer Webseite an eine andere: Quellseite besitzt HREF mit " ...?....." ruft Zielseite mit diesem HREF auf Zielseite wurde von Quellseite aufgerufen liest den Teil von HREF hinter dem ? als Zeichenkettendaten
.shape	Form/Gestalt eines Objektes siehe auch Eigenschaft .coords
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes



.tagUrn	Uniform Ressource Name (URN) laut Namensraum laut XMLNS-Attribut
.target	Name des Ziel-Fenster bzw. Ziel-Frame
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	MIME-Typ des Objektes (Multipurpose Internet Mail Extension) wenn die Eigenschaft .classid nicht kodiert wird, dann immer .type kodieren bzw. die im Browserstandard enthaltenen MIME verwenden Hinweis: MIME wird von Simple Mail Transfer Protocol (SMTP) benutzt z.B. bei Audio, Images, Video, Texten
.uniqueID	durch den Browser automatisch generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.urn	Uniform Ressource Name (URN) des Dokumentes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM- Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar



	DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
.insertAdjacentElement()	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.insertAdjacentHTML()	DOM wird geändert HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur
.releaseCapture()	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.replaceAdjacentText()	DOM wird nicht geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.scrollIntoView()	DOM wird nicht geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird
.setActive()	Objekt muss an sich schon renderbar sein Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen



	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt
	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
	Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft.
	dient
	Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
	nur sichtbar wenn Endtag geparkt
	DOM wird geändert

4.3.2.4.3.2. document.anchors Collection des Internet Explorer (HTML-Element A (Anker))

Feld der Zeiger aller Anker im Dokument

Feldelementefolge laut HTML-Coding

Erzeugung:

durch den Browser

Beispiel für Anker:

```
<A      ID="ID_Anker"
      HREF="url"
      NAME="logischer_anker_name"
      TARGET="logischer_window_name"
>
      anker_text
</A>
```

Hinweis zu HREF= :

kann leer sein	per	Leerkette ""
	oder	"javascript:void(0);"

zu Anspringen eines Ankers:

den Wert laut NAME ablegen in	
window.location.hash	ohne vorgesetztes #
window.location.href	mit vorgesetztem #

Syntax:

```
[ var ZeigerAufFeld = ] document.anchors
[ var ZeigerAufFeldElement = ] document.anchors[Index, [ SubIndex]]
```

Index	oder	Integer ab 0
		String Name oder ID des Elementes
		muss in [] kodiert sein
SubIndex		optional
		nur kodieren wenn Index ein String ist
		Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement.eigenschaft

ZeigerAufFeldElement.methode()

document.all.ID_Anker.eigenschaft

document.all.ID_Anker.methode()

Beispiel: Alle Textmarken (Anker) eines Dokumentes für deren Anwahl in einem Extra-Fenster anzeigen

```
<HTML>
<HEAD>
  <SCRIPT>
  <!--
    function textmarken_fenster_anzeigen()
    {
      var textmarken_fenster = open( "",
        "Textmarken-Fenster",
        "toolbar=0,location=0,scrollbars=0,menu=0,"
        + "dependent,resizable,status,width=150,height=300"
      );
      with(textmarken_fenster.document)
```



```

        {
            // HTML-Fenster formatieren
            window.name = "logischer_fenster_name"
            open("text/html")
            writeln("<HTML><HEAD>")
            writeln(    "<TITLE>Steuerung f&uuml;r &quot;"
                        + document.title
                        + "&quot;</TITLE>"
                    )
            writeln(    "<BASE HREF="
                        + location.href
                        + "' TARGET='logischer_fenster_name'>"
                    )
            writeln("</HEAD><BODY>")

            // Textmarken (Anker im Fenster anzeigen)
            for(var i = 0; i < document.anchors.length; i++)
            {
                writeln(    "<P><A HREF='#"
                            + document.anchors[i].name
                            + "'>"
                            + document.anchors[i].name
                            + "</A></P>"
                        );
            }

            writeln("</BODY></HTML>");
            close();
            focus();
        }

// -->
</SCRIPT>
</HEAD>
<BODY>
    <SCRIPT>
    <!--
        document.write(    "<P><A HREF='javascript: textmarken_fenster_anzeigen ();'>"
                        + "Textmarken-Fenster anzeigen</A></P>"
                    );

// -->
</BODY>
</HTML>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.3. applet Objekt des Internet Explorer

Objekt zum Einbinden ausführbaren Codes auf Basis einer Javamaschine (Virtuelle Javamaschine).

Applet muss alle Eigenschaften und Methoden, die im HTML-Dokument benutzt werden sollen, als **public** deklarieren.

Java-Applet (*.class) in HTML einbinden:

Beispiel

```

<APPLET
    ID="ID_Applet"
    CODE="class_datei"
    CODEBASE="quellverzeichnis"
    HEIGHT=applet_fenster_hoehe_in_pixel
    WIDTH=applet_fenster_breite_in_pixel
    MAYSCRIPT=true oder false
    ALT="alternativer_text"
    ALIGN=ausrichtung
    HSPACE=fenster_abstand_links_und_rechts_von_umgebung
    VSPACE=fenster_abstand_loben_und_unten_von_umgebung
>
    <PARAM Name="parameter_name"
        VALUE=parameter_wert

```




```

>
.....
</APPLET>

```

Zugriff:

ID_Applet.eigenschaft

ID_Applet.methode()

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
.altHTML	HTML-Script, der ausgeführt wird, wenn Objekt nicht geladen werden kann
.archive	Zeichenkette für Archive-Funktionalität eines Objektes
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.code	Url der *.class-Datei (kompilierter Javacode)
.codeBase	Url der Komponente für Dowload der Komponente vom Server auf den Client optionale Versionsprüfung möglich (nicht bei Applet-Komponente) um unnötigen Download zu ersparen: vorheriger Ableich der Version der Komponente auf Server und Client auch wenn der Versionsabgleich keinen Download ergab, wird immer HTTP Header-Transaktion ausgelöst
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.object	Zeiger auf das Objekt laut Applet bzw. laut Objekt object
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeigt, .offserLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag



	wirksam mit parsen des Ende-Tag
	nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt
	nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
STYLE	direkt im HTML-Element kodierter Style (Inline-Style)
	Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
	für Anspringen des Dokumentes
	Anspringen verbunden mit Focus erhalten
	--> Ereignisse werden ausgelöst !!
	unter IE 5.x
	onblur, onfocus
	ab IE 5.x
	onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste
	für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste
	für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
	vor IE 5.0 TABINDEX-Attribut muss kodiert sein
	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen
	außer ID, STYLE und per Script definierte Attribute
	Script-erzeugte Attribute nicht entfernen
	DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus
	manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern
	DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt



	auch für CSS-Layout
	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.detachEvent()	DOM nicht geändert Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.hasChildNodes()	DOM nicht geändert prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
.insertAdjacentElement()	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.namedRecordset()	DOM wird geändert Zeiger desjenigen Datenquellen-Record-Objektes mit Namen, das den Standard-Record-Set enthält Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an Eigenschaft .recordset liefert den Zeiger für den Standard-Record-Set in einem Datasource-Objekt (DSO)
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)



.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein
.replaceAdjacentText()	DOM wird nicht geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.scrollIntoView()	DOM wird nicht geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.4. document.applets Collection des Internet Explorer

Feld der Zeiger aller Applet-Objekte im Dokument

Elementefolge laut HTML-Coding

Applet muss alle Eigenschaften und Methoden, die im HTML-Dokument benutzt werden sollen, als **public** deklarieren

Erzeugung:

durch den Browser

Java-Applet (*.class) in HTML einbinden:

```
<APPLET ID="ID_Applet"
  CODE="class_datei"
  CODEBASE="quellverzeichnis"
  NAME="Name_Applet"
  HEIGHT=applet_fenster_hoehe_in_pixel
  WIDTH=applet_fenster_breite_in_pixel
  MAYSCRIPT=true oder false
  ALT="alternativer_text"
  ALIGN=ausrichtung
  HSPACE=fenster_abstand_links_und_rechts_von_umgebung
  VSPACE=fenster_abstand_loben_und_unten_von_umgebung
>
  <PARAM Name="parameter_name"
    VALUE=parameter_wert
  >
  ....
</APPLET>
```

Syntax:

```
[ var ZeigerAufFeld = ] document.applets
[ var ZeigerAufFeldElement = ] document.applets[Index [, SubIndex]]
```

Index	Integer ab 0 muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

`ZeigerAufFeldElement.eigenschaft`
`ZeigerAufFeldElement.methode()`

`document.all.ID_Applet.eigenschaft`
`document.all.ID_Applet.methode()`



document.all.Name_Applet.eigenschaft
document.all.Name_Applet.methode()

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.5. area Objekt des Internet Explorer

Area-Tag

Scripting ab 4.x

siehe auch Objekt map

Erzeugung:

Beispiel 1:

```
<IMG ID="ID_Img"
SRC="test.gif"
USEMAP="#MapName"
>
<MAP ID="ID_Map"
NAME="MapName">
<AREA SHAPE="rect" COORDS="0,0,33,33" HREF="test1.gif">
<AREA SHAPE="circle" COORDS="90,33,3" HREF="test2.gif">
</MAP>
```

Beispiel 2:

```
<MAP
NAME="logischer_map_name"
<AREA ID="ID_Area"
NAME="name_des_verweissensitiven_bereiches_der_grafik"
COORDS="koordinaten_liste_des_bereiches"
HREF="url"
SHAPE= "rect"
oder "poly"
oder "circle"
oder "default"
TARGET="logischer_window_name"
onClick="eventhandler_1"
onMouseOut="eventhandler_2"
onMouseOver="eventhandler_3"
> .....
</MAP>

SHAPE: rect Rechteck
poly Vieleck
circle Kreis

COORDS: bei rect und poly "x1,y1,...,xn,yn"
circle "x,y,r"
x Spalte in Pixel
y Zeile in Pixel
r Radius in Pixel
```

Hinweis: mehrere AREA sind möglich

Zugriff:

ID_Map.eigenschaft
ID_Map.methode()

Eigenschaften:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste
bei Ausführung der Tastenkombination wird
das Sprungziel wird focussiert
die Sprungquelle defocussiert
das Focus-Ereignis ausgelöst

.alt vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
Alternativer Text als Tooltip

ATOMICSELECTION Selektierbarkeit des Objektes einstellen

.begin Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
siehe Objekt currTimeState und Behavior .style.time2

.canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf

.className Klassenreferenz, Klassenname

.coords Koordinaten eines Objektes in Abhängigkeit zur Eigenschaft .shape

.dir Umflussrichtung



.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich						
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2						
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes						
.hash	Teil des Wertes der Eigenschaft .href also Teil der Url als Anker Teil folgt direkt hinter dem Nummernkreuz # und ohne Nummernkreuz						
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2						
.hideFocus	Focussierbarkeit						
.host	Hostname und Port einer Location oder Url in der Form " hostname:port "						
.hostname	Hostname einer Location oder Url in der Form " hostname:port " kann Domain oder IP sein						
.href	Ziel-Url oder Anker als Sprungziel (z.B. <A>-Tag) siehe auch Eigenschaften .rel und .rev						
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);						
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.						
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich						
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes						
.isTextEdit	Erzeugbarkeit eines Textbereiches						
.lang	Sprache für Anzeige von Sonderzeichen etc.						
.language	Sprache für Script festlegen						
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes						
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes						
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker						
.nodeType	Knotentyp laut attributes Collection						
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)						
.noHref	HREF-Wirkung (Eigenschaft .href) ein/ aus bei Click auf den Link						
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)						
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)						
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth						
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)						
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)						
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar						
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar						
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde						
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM						
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie						
.parentTextEdit	Textbereich des Elternobjektes referenzieren						
.pathname	Datei und Pfad eines Objektes						
.port	Port einer Location oder Url in der Form " hostname:port " basierend auf dem aktuellen Protokoll laut Eigenschaft .protocol z.B. <table><tr><th>Standard-Ports</th><th>Protokoll</th></tr><tr><td>21</td><td>FTP</td></tr><tr><td>80</td><td>HTTP</td></tr></table>	Standard-Ports	Protokoll	21	FTP	80	HTTP
Standard-Ports	Protokoll						
21	FTP						
80	HTTP						
.previousSibling	Referenz auf das Vorgängerkind						
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern aktueller Status des Objektes beim Füllen des Objektes mit Daten						
.readyState	Namensraum laut XMLNS-Attribut						
.scopeName	Teil des Wertes des Eigenschaft .href Teil folgt direkt dem Fragezeichen wird als Querystring oder Formdata bezeichnet hat nichts mit der Suche auf Webseiten zu tun Hinweis: Fragezeichen-Anhang an der HREF dient zur Übergabe von String-Werten einer Webseite an eine andere:						
.search							



	<p>Quellseite besitzt HREF mit " ...?....." ruft Zielseite mit diesem HREF auf</p> <p>Zielseite wurde von Quellseite aufgerufen liest den Teil von HREF hinter dem ? als Zeichenkettendaten</p>
.shape	Form/Gestalt eines Objektes
.sourceIndex	siehe auch Eigenschaft .coords
STYLE	Index des Objektes in der Collection document.all direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.target	Name des Ziel-Fenster bzw. Ziel-Frame
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt



	auch für CSS-Layout
	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben
	Overbereich der Maus ist mehr als 1 Pixel gross
	beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
	DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde
	Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
	DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern
	DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
	DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
	DOM nicht geändert
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
	DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
	DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
	DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
	DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
	DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur



.releaseCapture()	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.6. bgsound Objekt des Internet Explorer

Hintergrundmusik (wave, midi, mp3 etc.) oder Videoclip-Sound	
per HTML-Element	ab IE 3.x
per Script ohne ID-Verwendung	ab IE 4.x
mit ID-Verwendung	ab IE 5.x

Sound ist solange aktiv laut Anzahl der Wiederholungen der Sounderzeugung,
bis zum Entladen des des Dokumentes
bis zur Stummschaltung (siehe unten)

Alternativen zum bgsound Objekt sind: .style.time2 Behavior
Windows Media Player 7.1

Hinweis: Wenn ein Fenster mit dem Unterstrichsymbol rechts oben in der Fensterecke minimiert wurde, dann ist es durch Windows in die Hintergrund-Prioritätenfolge eingeordnet worden und arbeitet im Hintergrund bzw. garnicht. Mit anschließendem Maximieren (Symbol in der Fensterleiste rechts oben) wird das Fenster wieder angezeigt und das geladene Dokument **erneut** geöffnet. Konsequenz daraus ist, dass dieses Maximieren einem Neustart des Dokumentes entspricht, auch wenn der Programmierer diesen Zustand nicht erwünscht. Sound, der z.B. mit Start des Dokumentes erzeugt wird, erklingt erneut mit Maximierung nach einer Fensterminimierung. Analogon ist die Fenstereneinungung im Browser durch z.B. Ein- und Ausblenden der Favoritenleiste. Dieses Verhalten des Fensters mit seinem Dokument ist aber **nicht identisch** mit dem Verhalten nach einer Fenstergrößenänderung z.B. durch Rahmenverschiebung (resize). Die Aktionen des Fensters und seines Dokumentes bezüglich Resize müssen programmiert werden, sind also nicht standardmäßig vorhanden - im Gegensatz zum Verhalten mit/nach Fensterminimierung/-maximierung per Symbole in der rechten oberen Ecke der Fensterleiste. Sollte ein Frameset minimiert werden, dann wird das für den gesamten Frameset getan (Frameset hat 1 Fenster für alle Frames gesamt), allerdings mit Maximierung wird auch das Frameset-Dokument neu geladen. Wichtig dabei sind für Zeiger-Bezüge der Frames auf den Frameset per parent-Zeiger, dass die Daten im Frameset-Dokument mit Maximierung initialisiert werden und somit ebenfalls Daten der Frames, die im Frameset-Dokument abgelegt wurden, also z.B. Daten, die Verhaltensweisen der Frames vor einer Änderung der Frames-Inhalte gespeichert haben.



Erzeugung:

```

.....
<BGSOUND
    ID="freies_id"
    SRC="url_oder_dateiname"
    LOOP="infinite" oder Anzahl der Wiederholungen ab 1
    BALANCE =Wert
    VOLUME=Wert
>
.....

```

Hinweise: Es sind **mehrere** BGSOUND-Objekte (mit verschiedenen ID's) in einem Dokument instanzierbar, die auch **parallel** arbeiten. Man beachte dabei unbedingt die **intensive** Timer-Ressourcennutzung der BGSOUND-Objekte. Es ist dringend anzuraten, das Objekt, dessen Eigenschaft .loop auf unendlich belegt wurde, per URLzuweisung an die Eigenschaft .src stumm zuschalten, sobald der Sound nicht mehr benötigt wird: Es wird eine leere Url (Leerkette) zugewiesen. Eine Anwendung der Stummschaltung ist auch in der Realisierung eine Playliste aus Sounds anhand des BGSOUND-Objektes möglich (Folge von URLzuweisungen, deren zeitlicher Abstand die jeweilige bekannte Sounddauer ist (per Anweisung window.setTimeout() lässt sich die Wiedergabezeit der Sounds verwalten)). Die Nutzung von Playlisten-Alternativen zum BGSOUND-Objekt ist nur dann nötig, wenn Steuerungselemente zur Media-Datei oder zur Lautstärkeinstellung, die über den Rahmen der windowseigenen gehen, genutzt werden sollen.

Jede URLzuweisung an die Eigenschaft .src bewirkt den automatischen Neustart der Medium-Wiedergabe per BGSOUND-Objekt mit den aktuellen Einstellungen der Objekt-Eigenschaften sowie den Einstellungen innerhalb der Media-Datei. Wurde .src mit einer Leerkette belegt, so wird kein Sound wiedergegeben (Stummschaltung ohne Veränderung der windowseigenen Lautstärkeregelung).

Für **jeden** Medientyp ist ein **eigenständiges** BGSOUND-Objekt zu kodieren. Werden verschiedene Medientypen wie z.B. Midi und mp3 per gemeinsamen BGSOUND-Objekt verwaltet, so **kann** es passieren, dass die aktuelle Reglereinstellung zum Medientyp (z.B. mp3/wave) der windowseigenen Lautstärkeinstellung für die Soundwiedergabe nicht korrekt so belassen wird, wie sie der User dort permanent eingestellt hat, sondern für die Wiedergabe des Medientyps unerwünscht verändert wird, egal ob die Eigenschaft .volume des BGSOUND-Objektes durch den Programmierer verändert wurde oder nicht.

Das Vorladen eines Sounds per Script geht nur per BGSOUND selbst, da es keine Funktion analog zu new Image() gibt.

BGSOUND muss mit VOLUME auf -10 erzeugt werden (also stumm), wobei die Zuweisung zu .src das Laden in den Browser-Cache bewirkt,

Zugriff:

z.T. erst ab IE 5.x
 <BGSOUND ID="freies_id">
 in Script dann **freies_id** verwenden per **freies_id.eigenschaft** etc.

Beispiele für LOOP-Varianten:

<BGSOUND SRC="file:///c:/test/wav/test.wav">	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=0>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=1>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=10>	genau 10 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=-1>	endlos

```

<HEAD>
<SCRIPT>
    function KanalVerteilung(Wert)
    { ID_bgsound.balance = Wert; }

    function GenauEinmal()
    {
        ID_bgsound.loop = 1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }

    function Endlos()
    {
        ID_bgsound.loop = -1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>

```




```

<BUTTON onclick="Endlos()"></BUTTON>
<BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(10)"></BUTTON>
<BUTTON onclick="KanalVerteilung (0)"></BUTTON>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-10;"
>Mute

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-7;"
>25% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_bgsound.volume=-5;"
>50% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-2;"
>75% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=0;"
>100% Volume
</BODY>

```

Beispiel für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
// ++++++ globale Variablen, die verändert werden können
var SoundUrl = "56sec.mid";
var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++ Routinen der Sekundenzählung
var SekundenZahler = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekundenzählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekundenzählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
    }
}

```



```

        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl,SoundFileDauerInSekunden)
{
    this.SoundFileUrl          = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
                                                // Timerzeit für Rekursion
    this.SoundFileBeendet      = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()

```



```

//      nötig.
//      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
//      Damit wird der Style-Wert permanent neu berechnet.
//      Damit visualisieren sich die DIV's permanent neu.

// Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
//      also dynamisch anzeigen
ID_DIV_Balken.style.setExpression( "width",
                                   "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                   );

// Sekundenzähler in der Eigenschaft innerText mit Ausdruck belegen,
//      also dynamisch anzeigen
ID_DIV_SekundenZahler.setExpression("innerText", "SekundenZahler.toString()");

// - - - Messlatte statisch anzeigen
ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
ID_DIV_MessLatte.innerText = "Der Sound dauert "
                             + SoundDauerInSekunden.toString()
                             + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + ' STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

                    );

    document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
                    + ' STYLE="color:hotpink;font-weight:bold"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

                    );

    document.write( ' <DIV ID="ID_DIV_MessLatte"'
                    + ' STYLE="color:white;background-color:gray"'
                    + '>'
                    + '</DIV>'

                    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ----- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

Beispiel für Start und Stop des Sounds:

Das Objekt bgsound besitzt **keine** Methoden und auch nicht für Start und Stop des Sounds. Folgendes Beispiel schafft aber Abhilfe:

```

function SoundObjektErzeugen(Kette,Wert)
{
    this.SoundFileUrl      = Kette;
    this.SoundFileDauerInSekunden = Wert;
}

```



```

function SoundStummInstanzieren()
{
    document.write( '<BSOUND ID="ID_BGSound"'
        + ' SRC="' + StummSoundObjekt.SoundFileUrl + '"'
        + ' LOOP="1"'
        + '>'
    ); // alternativ auch SRC-Attribut weglassen
}

var SoundEndlosAbspielenAktiv=false;

function SoundEndlosAbspielen()
{
    ID_BGSound.src= SoundObjekt.SoundFileUrl;
    ID_BGSound.loop='infinite';
}

function SoundStoppen()
{
    ID_BGSound.src= StummSoundObjekt.SoundFileUrl; // alternativ Leerkette zuweisen
    ID_BGSound.loop='1';
}

SoundDauerInSekunden = 88;
SoundUrl              = "sound/test88.mid";
StummSoundUrl         = "sound/stumm.mid"; // leere MIDI-Datei ohne Daten, also ohne Tonwiedergabe

// +++++ Sound vorladen durch Erzeugung von Objekten
SoundObjekt = new SoundObjektErzeugen(SoundUrl,SoundDauerInSekunden);
StummSoundObjekt = new SoundObjektErzeugen(StummSoundUrl,0);
SoundStummInstanzieren();

// durch Aufruf von SoundEndlosAbspielen() wird der Sound erneut bzw. überhaupt gestartet
// durch Aufruf von SoundStoppen() wird der Sound gestoppt

```

Alternative zu BGSOUND:

Beispiel: Windows Media Player verwenden:

Achtung: Der Windows-Media-Player benutzt **sehr intensiv** die Timer von Windows, so dass parallele Rekursionen z.B. per setTimeout() nicht kontinuierlich ablaufen können !

```

function Y_SoundEndlosAbspielen()
{
    document.write( '<EMBED NAME="ID_WindowsMediaPlayer"'
        + ' SRC="' + SoundObjekt.SoundFileUrl + '"'
        + ' LOOP="infinite"'
        + ' STYLE="width=0px;height=0px"'
        + '>'
    );
}

SoundDauerInSekunden = 88;
X_SoundUrl           = "sound/test88.mid";

// +++++ Sound vorladen durch Erzeugung von Objekten
SoundObjekt = new SoundObjektErzeugen(SoundUrl,SoundDauerInSekunden);

```

Hinweis: Durch Veränderung von Style-Angaben width und height erscheint das Fenster des MediaPlayer inklusive der Bedienungsknöpfe, die je nach Medium aktivierbar sind.
Werden die Style-Angaben width bzw. height **nicht** angegeben, so erscheint der Windows Media Player im Standardfensterumfang.

Per **ID_WindowsMediaPlayer** sind Eigenschaften und Methoden des Player nutzbar.

Beispiel: Sound anstelle bg sound Objekt mit Timeline erzeugen:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }

```



[illegible]

Pendant des NS zum BGSOUND des IE:

per EMBED-Tag
immer im BODY kodieren

Beispiel:

```
<BODY>
.....
<EMBED SRC="url_oder_dateiname"
AUTOSTART=true
LOOP=true
HIDDEN=true
HEIGHT=0
WIDTH=0
>
</BODY>
```

Hinweise:

LOOP=true	endlos, sonst ab 1; für 1 auch false kodierbar
HIDDEN=true	Soundplayer-Fenster nicht anzeigen
HEIGHT=0 und WIDTH=0	Soundplayer-Fenster ohne Dimension

Eigenschaften:

.balance	Balance
	-10 bis +10
	-10 ganz links
	+10 ganz rechts
	0 genau mittig, default

Beispiel:

```
<HEAD>
<SCRIPT>
    function KanalVerteilung(Wert)
    { ID_bgsound.balance = Wert;}

    function GenauEinmal()
```



```

    {
        ID_bgsound.loop = 1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>
    <BUTTON onclick="Endlos()"></BUTTON>
    <BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung(10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung (0)"></BUTTON>
    <B>Volume control:</B>&nbsp;
    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-10;"
    >Mute

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-7;"
    >25% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
        onpropertychange="ID_bgsound.volume=-5;"
    >50% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
    >75% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
    >100% Volume
</BODY>

```

.disabled

Interaktionsfähigkeit

.id

nur wenn sichtbar so User-Interaktion möglich

.isDisabled

Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)

Interaktionsfähigkeit

nur wenn sichtbar so User-Interaktion möglich

.loop

Anzahl der Wiederholungen

nur Objekt BGSOUND bzw. IMG mit Sound bzw. INPUT-Element mit Sound

-1 endlos

0 genau 1 mal

> 0 Anzahl der Wiederholungen

Standard ist 1

1 entspricht 0

Beispiel:

```

<HEAD>
<SCRIPT>
function KanalVerteilung(Wert)
{ ID_bgsound.balance = Wert; }

function GenauEinmal()
{
    ID_bgsound.loop = 1;
    ID_bgsound.src = ID_bgsound.src; // restart
}

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>

```



```

<BUTTON onclick="Endlos()"></BUTTON>
<BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(10)"></BUTTON>
<BUTTON onclick="KanalVerteilung (0)"></BUTTON>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-10;"
>Mute

< INPUT TYPE="radio" NAME ="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-7;"
>25% Volume

< INPUT TYPE="radio" NAME ="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_bgsound.volume=-5;"
>50% Volume

< INPUT TYPE="radio" NAME ="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-2;"
>75% Volume

< INPUT TYPE="radio" NAME ="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=0;"
>100% Volume
</BODY>

```

aktueller Status des Objektes beim Füllen des Objektes mit Daten

Url der Daten

zulässige Sound-Dateisuffixe sind z.B. wav, mid (laut MIME-Typen des Internet Explorers)

aktuelle Wiedergabe-Lautstärke (Run time volume) per Objekt bgsound, wenn die Media-Datei nicht selbst Lautstärke-Einstellungen während der Wiedergabe vornimmt:

Bsp.:	MIDI	Volume ohne Wirkung
	WAVE	Volume soll Wirkung haben

Hinweis: eventuelle Veränderung der Regler in der Windows-Lautstärke-Regelung beachten.

-10 bis 0
0 maximal laut

Beispiel:

```
<HEAD>
<SCRIPT>

function KanalVerteilung(Wert)
{ ID_bgsound.balance = Wert; }

function GenauEinmal()
{
    ID_bgsound.loop = 1;
    ID_bgsound.src = ID_bgsound.src; // restart
}

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}

</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>

<BUTTON onclick="GenauEinmal()"></BUTTON>
<BUTTON onclick="Endlos()"></BUTTON>
<BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(0)"></BUTTON>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-10;"
>Mute

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-7;"
>25% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_bgsound.volume=-5;"
>50% Volume
```



```

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
        >75% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
        >100% Volume
</BODY>

```

Methoden:

keine

4.3.2.2.4.3.7. document.body Objekt des Internet Explorer

Rumpf des HTML-Dokumentes

Im Dokument gibt es genau 1 Rumpf also genau 1 Body Objekt

Die Ausführung von document.write() bzw. document.writeln(), das **HTML-Tags** erzeugt, hat 2 Konsequenzen und zwar genau dann, wenn diese Anweisung **nach dem kompletten Laden des Dokumentes, also nach Auslösung des Ereignisses onload** aktiviert wird:

Fakt 1:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen HTML-Dokumentes**, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

Die Methoden write() und writeln() erzeugen einen Datenstrom aus HTML-Elementen und das neue Dokument empfängt diesen so, als würde er aus einer HTML-Datei stammen. Mit Ende des Datenstromes wird quasi ein Dateieinde erkannt und das neue Dokument löst das Ereignis onload aus. Damit wird aber das neue Dokument zum aktuellen Dokument, also im Fenster über dem des alten Dokumentes angezeigt. Da das Fenster des neuen Dokumentes automatisch erzeugt wurde (nicht per Methode open()), sind also die Standards für eine Fenstererzeugung verwendet worden. Damit hat das neue Dokument einen History-Eintrag. Der User kann nun mit diesem zwischen dem alten und neuen Dokument umschalten.

Fakt 2:

Im Falle der o.g. nachträglichen Veränderung des Dokumentes um HTML-Elemente per write() bzw. writeln() kennt das neue, automatisch geöffnete Dokument das alte Dokument nur **als Eltern**. Es muss also im neuen Dokument mit dem Zeiger auf die Eltern gearbeitet werden, wenn Daten und Routinen der Eltern benutzt werden sollen (siehe Objekt window bzw. Objekt document). Mit anderen Worten: Das neue Dokument muss dann komplett per Script erzeugt werden, denn dieser Zeiger lässt sich nur über Script ansprechen. Jedes geladene Dokument hat ansonsten seine eigenständige Umgebung.

Wird versucht, per document.write() ein weiteres BODY-Tag zu erzeugen, so wird ein neues Fenster geöffnet und ein leeres Dokument geladen, das damit aktiv wird, oder es erfolgt eine Fehlermeldung.

Erzeugung in HTML:

Beispiel:

```

<BODY ID="ID_Body" ....>
....
</BODY>

```

NS und IE **können** den BODY verschieden erzeugen:

Nur der Internet Explorer erzeugt **automatisch** ein BODY-Objekt, wenn dieses im Dokument nicht kodiert wurde, weil z.B. sämtliche HTML-Elemente im HEAD des Dokumentes per Script erzeugt wurden.

Der NS verlangt immer einen BODY-Teil und ignoriert Script-Anweisungen im HEAD des Dokumentes, die vor dem Parsen von </BODY> bereits HTML-Elemente erzeugen wollen (z.B. per document.write()).

Es können also **nur** nachträglich neue HTML-Elemente per Script im HEAD des Dokumentes erzeugt werden.

Zugriff:

```

document.all.body.eigenschaft
document.all.body.methode()
document.all.ID_Body.eigenschaft
document.all.ID_Body.methode()

```

Beispiel 1 für Ausdruck des aktuellen Dokumentes:

```

<BODY>
        <INPUT TYPE="button" VALUE="Drucken"onclick="javascript:self.print()">
</BODY>

```

Beispiel 2 für Dokument ohne linken und oberen Standard-Seitenrand:

```
<BODY LEFTMARGIN=0 TOPMARGIN=0>
```

Hinweis: Beim Netscape muss kodiert werden

```
<BODY MARGINWIDTH=0 MARGINHEIGHT=0>
```



Beispiel 3 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
// ++++++ globale Variablen, die verändert werden können
var SoundUrl = "56sec.mid";
var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++ Routinen der Sekundenzählung
var SekundenZahler = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekundenzählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekundenzählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();
    }
}

```



```

// Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
SekundenZaehlen_Start();

// Sound erzeugen und sofort starten durch Url-Zuweisung
ID_BGSound.src=SoundObjekt.SoundFileUrl ;
SoundObjekt.SoundFileBeendet=false;

// Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
SoundTimeoutID = setTimeout(
    "SoundAbspielen()",
    SoundObjekt.SoundFileDauerInMillisekundeSekunden
);
}
else
{
    // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

    // Sound zu Ende
    SoundObjekt.SoundFileBeendet=true;

    // Sekundenzähler stoppen
    SekundenZaehlen_Stop();

    // und Meldung
    var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
    var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
    alert(
        "Wiedergabe beendet\nUngenauigkeit des Timers = "
        + TimerUngenauigkeit1.toString()+ " Sekunden\n"
        + "also " + TimerUngenauigkeit2.toString()+ " Ticks pro Sekunde"
    );
}
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ----- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ----- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
        "SekundenZahler * PixelBreiteProBalkenErweiterung"
    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText = "Der Sound dauert "
        + SoundDauerInSekunden.toString()
        + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{

```



```

document.write('<BODY></BODY>');

document.write(  '<BGSOUND ID= "ID_BGSound" LOOP="0">'

document.write(    '<DIV   ID="ID_DIV_Balken"'
+                  'STYLE="background-color:lightblue"'
+                  '>'
+                  '</DIV>'
+                  '<BR>'
                  );

document.write(    '<DIV   ID="ID_DIV_SekundenZahler"'
+                  'STYLE="color:hotpink;font-weight:bold"'
+                  '>'
+                  '</DIV>'
+                  '<BR>'
                  );

document.write(    '<DIV   ID="ID_DIV_MessLatte"'
+                  'STYLE="color:white;background-color:gray"'
+                  '>'
+                  '</DIV>'
                  );

// ++++++ Sound initialisieren und starten mit Laden des Dokumentes

// ----- Sound-Objekt erzeugen anhand globaler Variablen
SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

// ----- Sound-Objekt wiedergeben
SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

Hinweise:

bezüglich der Erzeugung von HTML-Elementen per HTML oder Script:

HTML-Elemente können per HTML-Tag **oder** per Methode `.createElement()` erzeugt werden. Die Behandlung von per HTML-erzeugten Elementen kann von der per DOM-Methoden erzeugten **abweichen**, da DOM die Art der Erzeugungen **differenziert**. Für HTML-erzeugte Objekte gibt es **immer noch** eigene Methoden (Spezialfälle), obwohl das nicht nötig wäre (vermutlich wegen der schrittweisen Erweiterung des DOM auf alle Elemente im Dokument). Empfehlung: Man verwende - wenn möglich - **nur** Methoden, die HTML- **und** Script-erzeugte Elemente bearbeiten und erreicht dadurch die Vereinheitlichung im Quellcode.

`.innerHTML` ruft intern `.createElement()` auf

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem

Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !

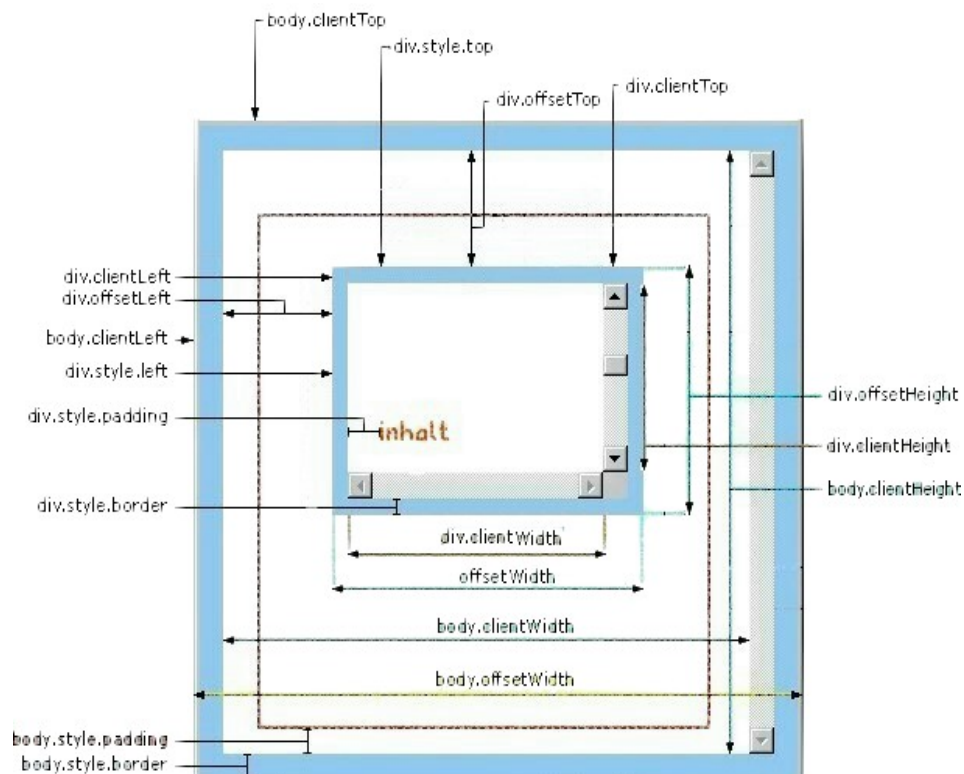
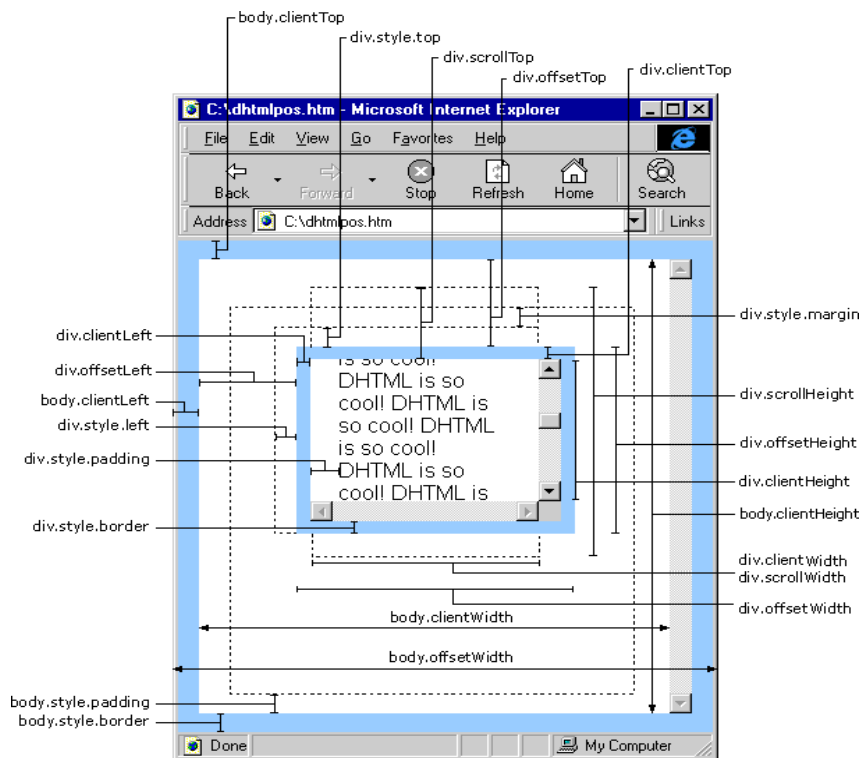
Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

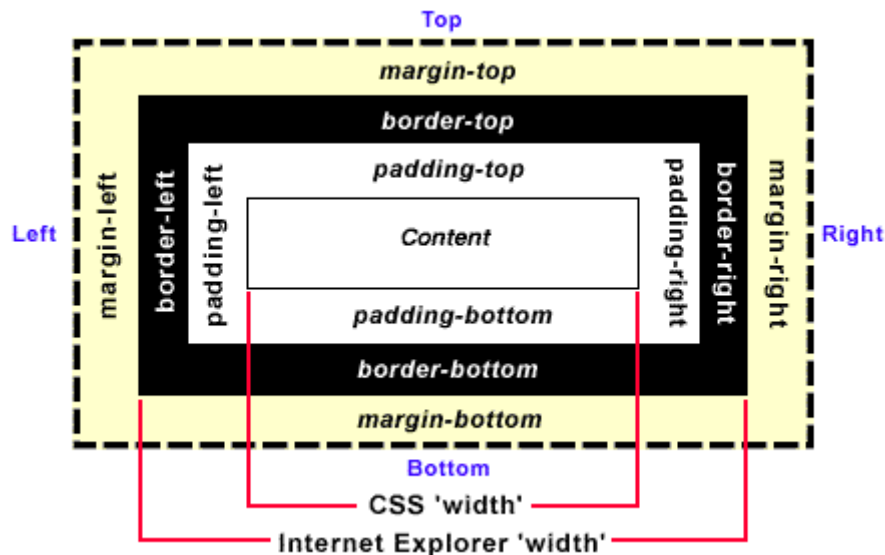
Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Defintion ersetzt wertmäßig den aktuellen Attributwert, wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).



Die Eigenschaften des IE als Grafiken:





Hinweis

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objekthinhaltes zum Aussenrand

4.3.2.2.4.3.7.1. Eigenschaften beim Internet Explorer

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.aLink	Farbe eines ALinks
ATOMICSELECTION	Selektierbarkeit einstellen
.background	Hintergrundbild
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen Hintergrundfarbe des Objektes bzw. Dokumentes
.bgProperties	Scroll-Eigenschaften des Hintergrundbildes beim Dokumentscrollen
.blockDirection	Umlauf um ein Objekt
.bottomMargin	Bottom Margin in Pixel
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umlflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.



.language	Sprache für Script festlegen
.leftMargin	Left Margin in Pixel des Dokumentes
.link	Farbe eines Links
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.noWrap	Wortumbruch einstellen
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von offsetHeight, offsetLeft, offsetTop und offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.rightMargin	Right Margin in Pixel des Dokumentes
.scopeName	Namensraum laut XMLNS-Attribut
.scroll	Scrollbar-Anzeige ein/aus vor IE 6.x nur BODY-Objekt ab IE 6.x alle HTML-Elemente wenn DOCTYPE im Kopf des Dokumentes kodiert
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.text	Textfarbe (Vordergrundfarbe)
.timeStartRule	deprecated Startpunkt der Timeline
.title	Tooltip-Text bei Mouse over über Objekt
.topMargin	Top Margin in Pixel des Dokumentes
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektierbarkeit z.B. mit Maus wird nicht an Kinder vererbt bei Wechsel: vorhergehende vorhandene Selektion wird nicht verändert
.vLink	Farbe eine VLINK



4.3.2.2.4.3.7.2. Methoden beim Internet Explorer

<code>.addBehavior()</code>	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
<code>.appendChild()</code>	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection <code>childNodes</code> angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
<code>.applyElement()</code>	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode <code>.createElement()</code> erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft <code>.innerHTML</code> gelöscht !
<code>.attachEvent()</code>	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode <code>.detachEvent()</code> Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
<code>.blur()</code>	Element den Focus wegnehmen und Event <code>onblur</code> auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 <code>TABINDEX</code> -Attribut muss kodiert sein ab IE 5.0 <code>TABINDEX</code> -Attribut muss nicht kodiert sein
<code>.clearAttributes()</code>	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
<code>.click()</code>	simuliert einen Klick auf das Element und löst <code>onclick</code> -Event aus manipuliert nicht den Focus
<code>.cloneNode()</code>	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout <code>onmouseover</code> -Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.contains()</code>	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
<code>.createControlRange()</code>	Selektionsbereich erzeugen es kann nur genau 1 Range pro Zeitpunkt existieren: wenn einer vorhanden, so diesen überschreiben
<code>.createTextRange()</code>	Textbereich erzeugen
<code>.detachEvent()</code>	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_</code> bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
<code>.doScroll()</code>	Click auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein !
<code>.dragDrop()</code>	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
<code>.fireEvent()</code>	ein Event auslösen
<code>.focus()</code>	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss <code>TABINDEX</code> -Attribut besitzen
<code>.getAdjacentText()</code>	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
<code>.getAttribute()</code>	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
<code>.getAttributeNode()</code>	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
<code>.getBoundingClientRect()</code>	Referenz auf <code>TextRectangle</code> -Objekt im Element holen
<code>.getClientRects()</code>	Referenz auf Feld der Zeiger auf <code>TextRectangle</code> -Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle



<code>.getElementsByTagName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.getExpression()</code>	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden <code>expression()</code> oder <code>setExpression()</code> zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
<code>.hasChildNodes()</code>	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.pause()</code>	deprecated auf Timeline pausieren
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu



	ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.resume()	DOM wird geändert deprecated Pause auf der Timeline beenden
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setCapture()	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.7.3. document.body.timeAll Collection des Internet Explorer

Feld der Zeiger aller per Behavior time2 verwalteten Elemente (getimte Elemente)
siehe auch style.time2 Behavior und .timeChildren Collection

Erzeugung:

durch den Browser

Syntax:

```
[ var ZeigerAufFeld = ] document.all.body.timeAll
[ var ZeigerAufFeldElement = ] document.all.body.timeAll [Index]

[ var ZeigerAufFeld = ] document.all.ID_Body.timeAll
[ var ZeigerAufFeldElement = ] document.all.ID_Body.timeAll [Index]
```

Index Integer ab 0
oder String Name oder ID des Elementes
muss in [] kodiert sein

ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

4.3.2.2.4.3.8. button Objekt des Internet Explorer

HTML-Container und Objekt-Prototyp eines Buttons z.B. auch Button in einem Formular oder Button per Objekt input
Wird Button innerhalb eines Formulars für das Senden benutzt, so
wird der Wert laut Eigenschaft .innerText gesendet
muss das NAME-Attribut für das Button ebenfalls kodiert sein wie für das Formular an sich !).

Zugriff:

Beispiel 1:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:VIDEO ID="ID_Video"
```



```

SRC="test.wmv"
STYLE="position:absolute;top:50px;height:100px"
>
</t:VIDEO>
<SPAN ID="ID_Span"
STYLE="position:absolute;top:165px;"
>
</SPAN>
<BUTTON ID="ID_Button"
onclick="ID_Span.innerText= ID_Video.abstract"
>
Klick
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
function LocationAendern()
{
    for(i=0;i<document.all.length;i++)
    {
        if(document.all(i).tagName=="IFRAME")
        {document.all(i).contentWindow.location = "http://www.test.com";}
    }
}
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="LocationAendern();">Location aendern</BUTTON>
<IFRAME SRC="http://www.test.de"></IFRAME>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT ID="ID_Script" FOR="ID_Botton" EVENT="onclick">
var Text1 = "Pferdchen hue !"
var Text2 = "Pferdchen brrr !"

if (ID_Botton.innerText == Text1)
{ID_Botton.innerText = Text2; }
else
{
    if (ID_Botton.innerText == Text2)
    { ID_Botton.innerText = Text1; }
}
</SCRIPT>
<BUTTON ID="ID_Botton" onmouseout="alert(ID_Script.event)">Hue oder Brrr ? </BUTTON>

```

Beispiel 4:

```

<SCRIPT>
function TesteMaus(Zeiger)
{
    if( ! Zeiger.contains(event.fromElement) )
    {alert("Maus über Button erkannt");}
}
</SCRIPT>
<BUTTON onmouseover=" TesteMaus(this)">Ueberfahre mich mit der Maus</BUTTON>

```

Beispiel 5:

```

<BUTTON>fokussierbar</BUTTON>
<BUTTON HIDEFOCUS="true">nicht fokussierbar</BUTTON>

```

Beispiel 6:

```

<P ID="ID_P">Dieser Text wird verändert</P>
<BUTTON onclick=" ID_P.innerText='Neuer Text'">Neuer Text</BUTTON>
<BUTTON onclick=" ID_P.innerText= Dieser Text wird verändert">Reset</BUTTON>

```

Beispiel 7:

```

<HEAD>
<SCRIPT>
function EditierbarkeitWechseln()

```



```

{
    var Editierbarkeit = ID_Span1.isContentEditable;
    var Editierbarkeit_Negation = ! Editierbarkeit;
    ID_Span1.contentEditable = Editierbarkeit_Negation;
    ID_Span2.innerText = Editierbarkeit_Negation;

    if (Editierbarkeit_Negation == false)
    { ID_Button.innerText="editierbar machen" }
    else
    { ID_Button.innerText="nicht editierbar machen" }
}
</SCRIPT>
</HEAD>
<BODY onload=" ID_Span2.innerText = ID_Span1.isContentEditable;">
    <BUTTON ID="ID_Button" onclick=" EditierbarkeitWechseln()">
        Klick mich
    </BUTTON>
    <SPAN ID="ID_Span1">Diesen Text editieren</SPAN>
    <SPAN ID="ID_Span2">Editierbarkeit</SPAN>
</BODY>

```

Beispiel 8:

```

<HEAD>
<SCRIPT>
    function KanalVerteilung(Wert)
    { ID_bgsound.balance = Wert; }

    function GenauEinmal()
    {
        ID_bgsound.loop = 1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }

    function Endlos()
    {
        ID_bgsound.loop = -1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>
    <BUTTON onclick="Endlos()"></BUTTON>
    <BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung(10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung (0)"></BUTTON>
    <B>Volume control:</B>&nbsp;
    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-10;"
    >Mute

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-7;"
    >25% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
        onpropertychange="ID_bgsound.volume=-5;"
    >50% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
    >75% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
    >100% Volume
</BODY>

```

Beispiel 9 für Sekundenbalken:

```

<HTML>
<HEAD>
<STYLE>

```



```

        BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

    function ZurueckSetzen()
    { Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"

    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"

    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"

```



```

        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P        STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!)



	muss beim Formular für alle zu sendenden Felder kodiert sein !!
	Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrolling	Scrollenbar erzeugen
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes



	siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Art des Buttons (Standard-Behavior des Buttons)
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.createTextRange()	Textbereich erzeugen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar



	DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByTagName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.hasChildNodes()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
.insertAdjacentElement()	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.insertAdjacentHTML()	DOM wird geändert HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.insertBefore()	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern



	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt
	ersetzende Objekt muss per Methode .createElement() erzeugt worden sein
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
	sichtbar erst mit parsen des Endetags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren
	aber ohne es zu fokussieren
	und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen
	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt
	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
	Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft.
	dient
	Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
	nur sichtbar wenn Endetag geparkt
	DOM wird geändert

4.3.2.2.4.3.9. comment Objekt des Internet Explorer

Kommentar-Objekt

Kommentar wird nicht angezeigt (gerendert)

Eigenschaften:

.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.data	Url der Daten des Objektes
.disabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
	Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.length	Anzahl der Zeichen im Kommentar (comment-Objekt), ab IE 6.x
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)
.offsetParent	Referenz der Eltern
	für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.outerText	Referenz auf den gesamten Plain-Text im Objekt
	nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie



.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.sourceIndex	Index des Objektes in der Collection document.all
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.text	Text eines Objektes
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.appendData()	String an das Ende des Objektes anhängen
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.deleteData()	Teilkette aus einem Objekt entfernen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann



	wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.insertData()	Teilkette in ein Objekt einfügen
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceData()	Teilkette in einem Objekt ersetzen
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.substringData()	Teilkette aus einem Objekt lesen
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.10. div Objekt des Internet Explorer

ab IE 4.x

Das DIV- bzw. SPAN-Objekt ähneln sich sehr stark. DIV kann ein automatisches
 am DIV-Ende implizieren.

Die übliche Zugriffsweise ist über das ID-Attribut (Eigenschaft .id) des DIV bzw. SPAN.

DIV bzw. SPAN können im Elternobjekt (Dokument im Fenster oder übergeordneter DIV bzw. SPAN) erzeugt werden,
als HTML-Element üblicherweise im BODY-Teil des Dokumentes



per Script durch z.B. `document.write("<DIV >...");`

Die Anzeige (das Rendern) des DIV bzw. SPAN erfolgt erst mit dem Parsen des Ende-Tags, also `</DIV>` bzw. ``. Danach können Komponenten zur Laufzeit nachträglich verändert werden, die aber nur z.T. sofort sichtbar werden (teilweise erst nach dem Reload des betreffenden Dokumentes).

NS und IE können das DIV- und SPAN-Objekt erzeugen, implementieren und rendern diese aber verschiedenartig.

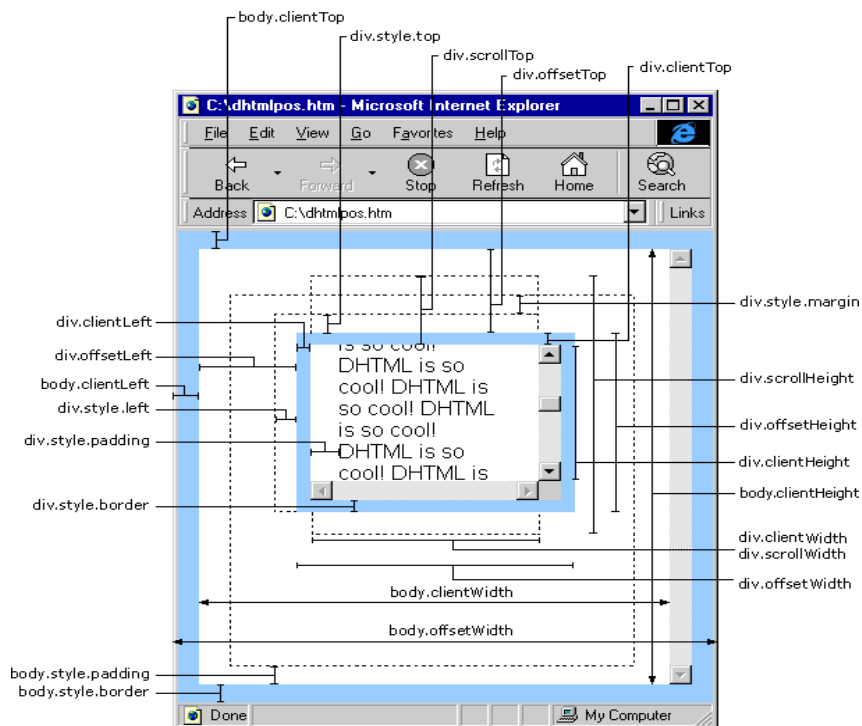
Beispiel: Nur der Internet Explorer erzeugt automatisch ein BODY-Objekt, wenn dieses im Dokument nicht kodiert wurde, weil z.B. sämtliche HTML-Elemente im HEAD des Dokumentes per Script erzeugt wurden.

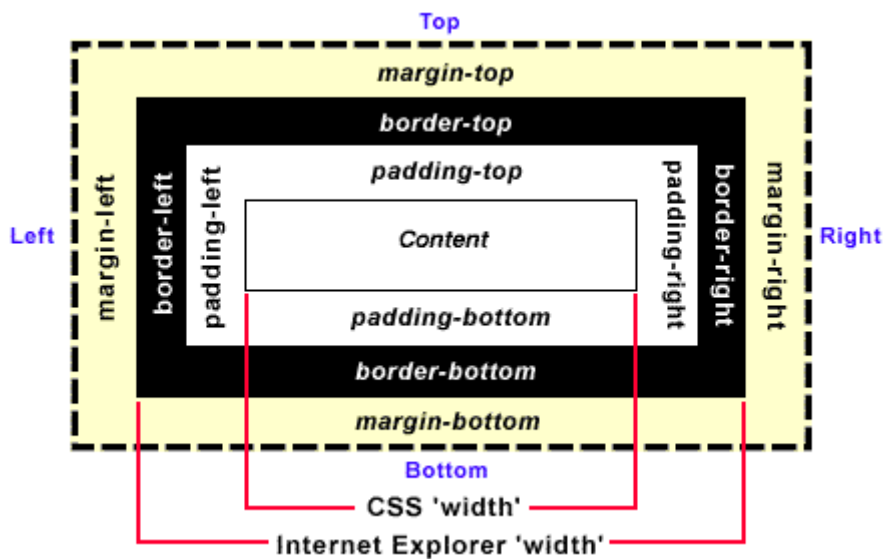
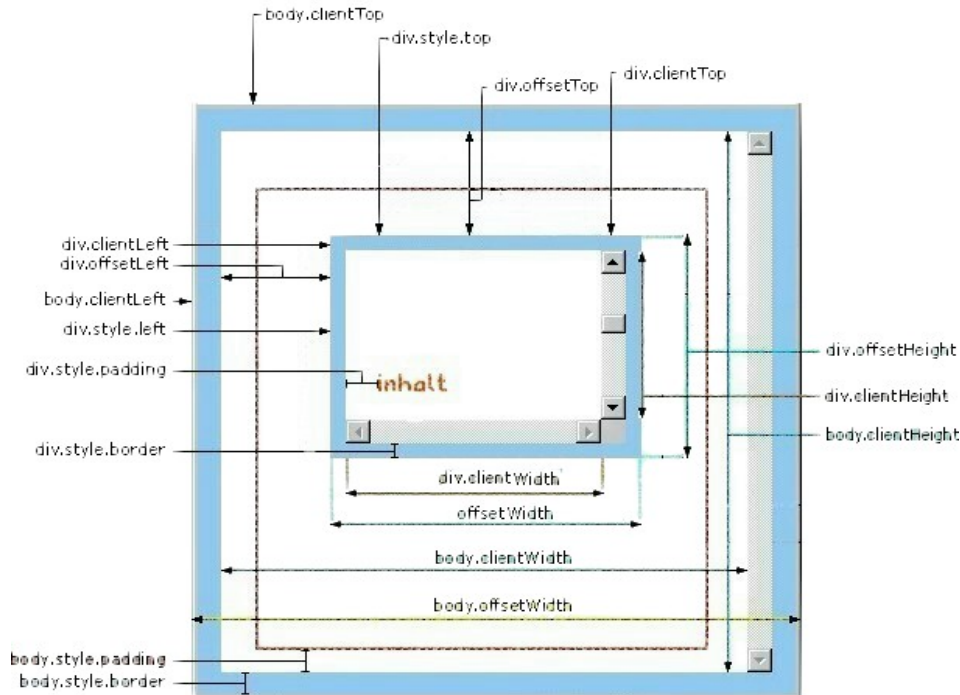
Der NS verlangt immer einen BODY-Teil und ignoriert o.g. Script-Anweisungen im HEAD des Dokumentes, solange der BODY des Dokumentes nicht komplett geparkt wurde (`</BODY>` noch nicht erkannt wurde). Es können also nur nachträglich neue HTML-Elemente per Script im HEAD des Dokumentes erzeugt werden. Scriptaufrufe aus dem BODY-Teil werden mit dem BODY geparkt, egal ob die Scripte im HEAD oder BODY liegen.

Im NS unter der Version 6.x wird anstelle von DIV- bzw. SPAN-Objekt das LAYER-Objekt favorisiert. Ab NS 6.x wird das Layer-Objekt radikal nicht mehr unterstützt: Es sind für die Ebenendarstellung (überlappende Objekte) keine Layer mehr sondern z.B. DIV oder SPAN verwendbar.

Bezüglich zeitsteuernder Eigenschaften/Methoden werden ab IE 6.x auch `Behavior.time2` und Objekt `currTimeState` benutzt.

Wichtige Eigenschaften im IE:





Hinweis Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !
 Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert, wenn gleiche Attributnamen / Eigenschaften betroffen sind (sonst hinzufügen).

Beispiele:



Beispiel 1:

```
<HTML>
<HEAD>
<SCRIPT>
    function HandlerFuerOnMoveStart()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "green";
        ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
    }

    function HandlerFuerOnMove ()
    {
        ID_Span1.innerHTML = event.srcElement.offsetLeft;
        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandlerFuerOnMoveEnd()
    {
        // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
        var ZeigerAufObjektMitEvent = event.srcElement;

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    // 2-D Positionierung einschalten
    document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offsetTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>
```

Beispiel 2 für Sound mit Sekundenanzeige:

```
<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln
    // Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    // ++++++ Routinen der Sekundenzählung
    var SekundenZahler = 0;
    var SekundenZahlerTimeoutID = null;

    function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
    {
        // Zähler erhöhen
        SekundenZahler++;
```



```

        // visuelle Anzeige im Dokument auffrischen, also alle Ausdrücke der Style-Werte
        // neu berechnen und damit alle DIV's neu visualisieren
        document.recalc();
    }

    function SekundenZaehlen_Start()
    {
        // prüfen ob Sekunden zählen nicht bereits läuft
        if (SekundenZählerTimeoutID == null)
        {
            // nicht aktiv

            // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
            SekundenZählerTimeoutID = setInterval("SekundenZaehlen()", 1000);
        }
    }

    function SekundenZaehlen_Stop()
    {
        // prüfen ob Sekunden zählen aktiv ist
        if (SekundenZählerTimeoutID != null)
        {
            // aktiv, also stoppen
            clearInterval(SekundenZählerTimeoutID);
            SekundenZählerTimeoutID = null;
        }
    }

    // ++++++ Routine zur Erzeugung Sound-Objekt
    function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
    {
        this.SoundFileUrl = SoundFileUrl;
        this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
        // Timerzeit für Rekursion
        this.SoundFileBeendet = true; // kein Sound aktiv
    }

    // ++++++ Routinen zur Wiedergabe Sound-Objekt
    var SoundTimeoutID=0;

    function SoundAbspielen()
    {
        // prüfen ob Sound nicht bereits aktiv ist
        if (SoundObjekt.SoundFileBeendet)
        {
            // Anzeige initialisieren
            SekundenAnzeigeInit();

            // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
            SekundenZaehlen_Start();

            // Sound erzeugen und sofort starten durch Url-Zuweisung
            ID_BGSound.src=SoundObjekt.SoundFileUrl ;
            SoundObjekt.SoundFileBeendet=false;

            // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
            SoundTimeoutID = setTimeout(
                "SoundAbspielen()",
                SoundObjekt.SoundFileDauerInMillisekundeSekunden
            );
        }
        else
        {
            // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

            // Sound zu Ende
            SoundObjekt.SoundFileBeendet=true;

            // Sekundenzähler stoppen
            SekundenZaehlen_Stop();

            // und Meldung
            var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZähler;

```




```

        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
                                        "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                                + SoundDauerInSekunden.toString()
                                + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BG SOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + ' STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

    );

    document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
                    + ' STYLE="color:hotpink;font-weight:bold"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

    );

    document.write( ' <DIV ID="ID_DIV_MessLatte"'
                    + ' STYLE="color:white;background-color:gray"'
                    + '>'
                    + '</DIV>'

    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

```




```

// ----- Sound-Objekt erzeugen anhand globaler Variablen
SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

// ----- Sound-Objekt wiedergeben
SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
<!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.blockDirection	Umfluss um ein Objekt
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end siehe Objekt currTimeState und Behavior .style.time2
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.noWrap	Wortumbruch einstellen
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)



.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scroll	Scrollbar-Anzeige ein/aus vor IE 6.x nur BODY-Objekt ab IE 6.x alle HTML-Elemente wenn DOCTYPE im Kopf des Dokumentes kodiert
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
SYSTEMLANGUAGE	Sprache festlegen für das Objekt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist siehe Objekt currTimeState und Behavior .style.time2
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn



	die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar
.click()	DOM wird geändert simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Click auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein !
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.hasFocus()	Objekt im Focus-Zustand
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert



<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
<code>.setExpression()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-



Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 .swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.11. document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)

Feld der Zeiger aller als Plugin per EMBED-Tag eingebetteten Objekte im Dokument (inkl. Applets)
 Elementefolge laut HTML-Coding

Achtung: Ab IE 6.x werden keine Plugins mehr unterstützt, damit ist die plugins Collection hinfällig. Inwieweit diese Collectionen noch implementiert ist oder bleibt, ist durch den Programmierer zu prüfen. Ab dem IE 6.x muss jedes Plugin, das in das Dokument eingebunden werden soll, durch ein ActiveX-Control implementiert werden. Plugins, wie sie beim Netscape existieren, laufen unter dem IE ab 6.x nicht mehr.

Es ist zu vermuten, dass im IE ab 6.x diese Collection nur eine Dummy-Funktion hat, also existiert, aber nie angefasst wird.

Diese Collection ist ein Alias für die plugins Collection **nur** bezüglich von Plugins (und nicht anderen einbettbaren Objekten), wobei letztere nur der Kompatibilität mit anderen plugin-fähigen Browsern dient. Alle eingebetteten Objekte haben in document.embeds ihren Zeiger (inklusive Plugins, Ausnahme: siehe tiefer).

Das Ansprechen von Plugins kann über die Collection navigator.mimeTypes per Eigenschaft .enabledPlugin erfolgen, die einen Zeiger auf das installierte Plugin enthält (wenn nicht installiert, so null.Zeiger). Diese Collection muss aber beim Internet Explorer nicht komplett gefüllt sein, kann aber (ausprobieren). Wenn der Zeiger nicht null ist, dann sind die plugin-eigenen Methoden und Eigenschaften über Punktnotation ansprechbar.

Es ist empfehlenswert, beim Internet Explorer **nur** mit der document.embeds Collection zu arbeiten. Die Collection navigator.mimeTypes kann zwar unter NS und IE mit dem gleichen Script-Code angesprochen werden, aber das ist spätestens ab IE 6.0 nicht mehr möglich.

Erzeugung:

durch Browser

Beispiel für EMBED-Tag beim IE unter 6.x und beim NS:

```
<EMBED
  SRC="url"
  TYPE="mime_typ"           // z.B. "image/gif"
  PLUGINSOURCE="plugin_url"
  HEIGHT="hoehe_plugin_objekt"
  WIDTH="breite_plugin_objekt"
  NAME="ID_Embed"
  HIDDEN="true oder false"  // true so Objekt unsichtbar
>
  <PARAM Name="parameter_name" VALUE=parameter_wert>
  ....
</EMBED>
```

Syntax:

```
[ var ZeigerAufFeld = ] document.embeds
[ var ZeigerAufFeldElement = ] document.embeds[Index, [SubIndex]]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement.eigenschaft
 ZeigerAufFeldElement.methode()

beim IE:

```
document.all.ID_Embed.eigenschaft
document.all.ID_Embed.methode()
```

beim NS:

```
document.ID_Embed.eigenschaft
document.ID_Embed.methode()
```

Eigenschaften beim Internet Explorer:

.length Anzahl der Feldelemente also Feldlänge

Methoden beim Internet Explorer:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
 Attributnamen (analog zu ID oder NAME-Attribut) liefern
 außer bei Formular mit <INPUT TYPE=image ...>



da dafür die children-Collection verwendet werden muss !!!

`.namedItem()` Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

`.tags()` Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

`.urns()` Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.12. fieldSet Objekt des Internet Explorer

Objekt dient zum Rendern einer Box z.B. um einen Text
als optische Gruppierung zusammengehöriger Texte

Eigenschaften:

`.accessKey` Tastaturzugriff auf ein Objekt per Alt + Taste
bei Ausführung der Tastenkombination wird
das Sprungziel wird focussiert
die Sprungquelle defocussiert
das Focus-Ereignis ausgelöst
vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt

`.align` Ausrichtung

`ATOMICSELECTION` Selektierbarkeit einstellen

`.begin` Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft `.timeAction`
siehe Objekt `currTimeState` und Behavior `.style.time2`

`.blockDirection` Umfluss um ein Objekt

`.canHaveChildren` prüfen ob Kind möglich ist, also ob Objekt Parent sein kann

`.canHaveHTML` prüfen ob Objekt HTML-Tags enthalten darf

`.className` Klassenreferenz

`.clientHeight` Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
ohne Rahmen
ohne Scrollbalken

`.clientLeft` Abstand in Pixel zum linken Rand des Fensters

`.clientTop` Abstand in Pixel zum oberen Rand des Fensters

`.clientWidth` Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
ohne Rahmen
ohne Scrollbalken

`.contentEditable` Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat)
Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.

`.dataFld` Datenquelle-Name vergeben (ID)

`.dir` Umflussrichtung

`.disabled` Interaktionsfähigkeit
nur wenn sichtbar so User-Interaktion möglich

`.end` Objektaktivitäten laut Eigenschaft `.timeAction` beenden
ab IE 6.x
alternativ: Eigenschaft `.dur`
siehe Objekt `currTimeState` und Behavior `.style.time2`

`.firstChild` Zeiger auf das ERSTE Kind laut `childNodes`-Collection eines Objektes

`.form` Zeiger auf das Formular (Formular als Container)
ab IE 6.x für Elemente `fieldSet`, `label`, `legend`

`.hasMedia` Objekt ist HTML-Media-Objekt
siehe Objekt `currTimeState` und Behavior `.style.time2`

`.hideFocus` Focussierbarkeit

`.id` Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)

`.innerHTML` Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
ein Tag ohne Ende-Tag kann kein `innerHTML` haben, z.B. ``
dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
also nach dem Laden des Objektes
aber wirksam erst mit parsen des HTML-Endetag

`.innerText` Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
ein Tag ohne Ende-Tag kann kein `innerHTML` haben, z.B. ``
dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
also nach dem Laden des Objektes
aber wirksam erst mit parsen des HTML-Endetag

`.isContentEditable` Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.

`.isDisabled` Interaktionsfähigkeit
nur wenn sichtbar so User-Interaktion möglich

`.isMultiLine` Mehrzeiligkeit des Objekthinhaltes

`.isTextEdit` Erzeugbarkeit eines Textbereiches

`.lang` Sprache für Anzeige von Sonderzeichen etc.

`.language` Sprache für Script festlegen

`.lastChild` Zeiger auf das LETZTE Kind laut `childNodes` collection eines Objektes

`.nextSibling` Zeiger auf das NACHFOLGENDE Kind laut `childNodes` collection eines Objektes

`.nodeName` String als Name des Kindes (Knoten, Node, Element)
also TAG-Bezeichner, Attribut-Name; #text für Anker

`.nodeType` Knotentyp laut `attributes` Collection

`.nodeValue` Knotenwert (Wert des Kindes, Node, Elementes)
nur für Text- und Attribut-Elemente



	nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von offsetHeight, offsetLeft, offsetTop und offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektierbarkeit z.B. mit Maus wird nicht an Kinder vererbt bei Wechsel: vorhergehende vorhandene Selektion wird nicht verändert

Methoden:

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen



	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbare DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut):



	siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird



.setActive()	Objekt muss an sich schon renderbar sein Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.13. font Objekt des Internet Explorer

Implementation des Font mit seinen Eigenschaften

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz
.color	Farbe des Objektes
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.face	Familie des Font-Typs (Font-Face) z.B. Courier
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes



.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.size	Fontgröße
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde



<code>.applyElement()</code>	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode <code>.createElement()</code> erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft <code>.innerHTML</code> gelöscht !
<code>.attachEvent()</code>	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode <code>.detachEvent()</code> Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
<code>.blur()</code>	Element den Focus wegnehmen und Event <code>onblur</code> auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 <code>TABINDEX</code> -Attribut muss kodiert sein ab IE 5.0 <code>TABINDEX</code> -Attribut muss nicht kodiert sein
<code>.clearAttributes()</code>	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
<code>.click()</code>	simuliert einen Klick auf das Element und löst <code>onclick</code> -Event aus manipuliert nicht den Focus
<code>.cloneNode()</code>	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout <code>onmouseover</code> -Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.contains()</code>	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
<code>.detachEvent()</code>	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_bezeichnung</code> zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
<code>.dragDrop()</code>	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
<code>.fireEvent()</code>	ein Event auslösen
<code>.focus()</code>	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss <code>TABINDEX</code> -Attribut besitzen
<code>.getAdjacentText()</code>	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
<code>.getAttribute()</code>	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
<code>.getAttributeNode()</code>	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
<code>.getBoundingClientRect()</code>	Referenz auf <code>TextRectangle</code> -Objekt im Element holen
<code>.getClientRects()</code>	Referenz auf Feld der Zeiger auf <code>TextRectangle</code> -Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein <code>Rectangle</code>
<code>.getElementsByTagName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei <code>document.all</code> Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementsById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.getExpression()</code>	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden <code>expression()</code> oder <code>setExpression()</code> zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout



	(nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert



.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.14. document.form Objekt des Internet Explorer

Dieses Objekt dient der formular-orientierte Datenbeschaffung zum Zweck der Datenübersendung an den Server. Es ist möglich, mit einem Formular versteckt Daten zu transferieren, die dem User nicht angezeigt bzw. bewusst gemacht werden. Der Programmierer sollte bei der Nutzung von Scripten in Formularen die Sorgfaltspflicht bezüglich Userdaten umsetzen. Im Internet Explorer ist dem User das Abschalten der Autovervollständigung auch bezüglich Formulare anzuraten.

Das input Objekt ist der Container für alle Input-Varianten im Formular. Das Objekt img ist der Container für input image. Das Objekt button ist der Container für input button. Das Objekt textarea ist der Container für input textarea. Das Objekt select ist der Container für input option und input select. Container vererben ihre Eigenschaften an die Formularelemente.

Erzeugung unter HTML:

Beispiel:

```
<FORM ID="ID_Formular"
NAME="logischer_formular_name" // muß für alle zu sendenden Formular-Elemente ebenfalls
// kodiert sein, wenn die Elemente Daten des
// Formulars darstellen
TARGET="zielfenster" // Ausgabefenster z.B. des CGI-Scripts für
ACTION="auswertungs_url" // Auswertung und Antwort auf abgeschicktes Formular
METHOD="get" oder "post"
ENCTYPE="mime_typ" // z.B. text/* für E-Mail
onreset="eventhandler1" // erzeugt Event onreset
// Reaktion VOR dem Rücksetzen ist zu programmieren per
// Eventhandler:
// muß return-Anweisung haben:
// return true; so wird Formular auch
// zurückgesetzt
// return false; so wird Formular nicht
// zurückgesetzt
// Das Rücksetzen an sich macht der Browser per
// Methode .reset()

onsubmit="eventhandler2" // erzeugt Event onsubmit
// Reaktion VOR dem Senden ist zu programmieren per Eventhandler:
// muß return-Anweisung haben:
// return true; so wird Formular auch
// abgesendet
// return false; so wird Formular nicht
// abgesendet
// Das Absenden an sich macht der Browser per
// Methode .submit()

>
formularinhalt
</FORM>
```

Formularinhalte können diverse HTML-Elemente sein., vorallem Elemente, die Daten speichern können wie z.B. TEXTAREA, SELECT, OPTION etc.. Ein sehr beliebtes HTML-Element ist das input Objekt, das in diversen Varianten kodiert werden kann.

Beispiele:

Beispiel 1:

```
<HTML>
<FORM ACTION="http://www.test.de/sample.asp" METHOD="POST">
<SELECT NAME="Flavor">
<OPTION VALUE="Test1"> Test1
<OPTION VALUE="Test2"> Test2
<OPTION VALUE=" Test3" SELECTED> Test3
</SELECT>
<INPUT TYPE=SUBMIT>
</FORM>
</HTML>
```

Beispiel 2:




```

<FORM ACTION="mailto:test@test.de" METHOD=GET>
  <INPUT NAME=subject TYPE=hidden
    VALUE="Testt%20Product%20Information%20Anforderung"
  >
    volle Mailadresse eingeben
  <BR>
  <TEXTAREA NAME=body COLS=40></TEXTAREA>
  <INPUT TYPE=submit VALUE="absenden "
</FORM>

```

Beispiel 3:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
  // Cachename festlegen
  //      es können diverse Cachennamen definiert und somit Versionen von Cache
  //      verwaltet werden
  var FreierCacheName = "InputCache";

  // Cache-Attribut festlegen
  //      es können diverse Attribute definiert und somit Versionen von Input-Daten
  //      verwaltet werden
  var FreiesCacheAttribut = "InputCacheAttribut";

  // zu cachende Daten referenzieren
  var InputDatenObjekt = ID_Formular.ID_Input;

  function InputSichern()
  {
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;

    // Attribut instanzieren und mit Daten füllen
    InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

    // und Cache saveen
    InputDatenObjekt.save(FreierCacheName);
  }

  function InputLaden()
  {
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
  }

</SCRIPT>
</HEAD>
<BODY>
  <FORM ID="ID_Formular">
    <INPUT ID="ID_Input"
      CLASS="user_data_speicher_klasse"
      TYPE="text"
    >
    <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">

```



```

        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML>
<HEAD>
<SCRIPT>
    function EventHandler_AktionVorReset()
    {
        // ein Hinweis im Textarea anzeigen
        ID_Textarea.value += "Formular zuruecksetzen ???";

        // wirklich rucksetzen ???
        return( confirm("Wirklich rucksetzen ?"));
        // true so Reset durch Browser ausfuehren lassen
        // false so kein Reset durch Browser
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV>
        <FORM NAME="Formular" onreset="EventHandler_AktionVorReset();">
            <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
            <BR>
            <BUTTON onclick="form.reset();">OnReset ausloesen</BUTTON>
            <BR><BR>
            <INPUT TYPE="reset" VALUE="oder hiermit zuruecksetzen"
                onclick="form.reset()"
            >
        </FORM>
    </DIV>
    <TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>

```

Beispiel 5:

```

<HTML>
<HEAD>
<SCRIPT>
    function EventHandler_AktionVorSubmit()
    {
        // ein Hinweis im Textarea anzeigen
        ID_Textarea.value += "Formular senden ???";

        // wirklich senden ???
        return( confirm("Wirklich senden ?"));
        // true so Submit durch Browser ausfuehren lassen
        // false so kein Submit durch Browser
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV>
        <FORM NAME="Formular" ACTION=" ..." METHOD=" "
            onsubmit="EventHandler_AktionVorSubmit();">
            <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
            <BR>
            <BUTTON onclick="form.submit();">OnSubmit ausloesen</BUTTON>
            <BR><BR>
            <INPUT TYPE="submit" VALUE="oder hiermit senden"
                onclick="form.submit()"
            >
        </FORM>
    </DIV>
    <TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>

```

Beispiel 6:

```

<FORM ACTION="test.htm">
    <INPUT TYPE="submit" VALUE="Gehe zu test.htm">
</FORM>

```



Zugriff:

Hinweise: Anstelle des logischen Namens laut NAME-Attribut kann auch der Wert des ID-Attributes verwendet werden.
Das NAME-Attribut muss kodiert werden, wenn Daten des Formularelementes zu senden sind per .submit()
(zu kodieren im Formular selbst und für jedes Formular-Element, das Daten senden soll).

auf das Formular:

beim NS:

```
logischer_formular_name.eigenschaft  
logischer_formular_name.methode()
```

```
document.forms[index].eigenschaft  
document.forms[index].methode()
```

index: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

beim IE:

```
document.all.logischer_formular_name.eigenschaft  
document.all.logischer_formular_name.methode()
```

```
document.forms[index].eigenschaft  
document.forms[index].methode()
```

index: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

auf ein Formular-Element:

Formularinhalte können diverse HTML-Elemente sein., vor allem Elemente, die Daten speichern können wie z.B. TEXTAREA, SELECT, OPTION etc.. Ein sehr beliebtes HTML-Element ist das input Objekt, das in diversen Varianten kodiert werden kann. **Jedes Element** innerhalb von <FORM> .. </FORM> wird zum Kind des Objektes form. Daher ist Punktnotation nötig, um den Formularinhalt, also die Kinder des Objektes form, referenzieren zu können.

beim NS:

```
logischer_formular_name.logischer_formular_element_name.eigenschaft  
logischer_formular_name.logischer_formular_element_name.methode()
```

```
var ZeigerAufFormular = document.forms[index];  
ZeigerAufFormular.logischer_formular_element_name.eigenschaft  
ZeigerAufFormular.logischer_formular_element_name.methode()
```

index: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

```
var ZeigerAufFormular = document.forms[index1];  
ZeigerAufFormular.elements[inde2].eigenschaft  
ZeigerAufFormular.elements[inde2].methode()
```

index1: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

index2: ab 0
Nummer des Formular-Elementes im Formular
muss in [] kodiert sein

```
var ZeigerAufFormular = document.forms[index1];  
var ZeigerAufFormularElement = ZeigerAufFormular.elements[index2];  
ZeigerAufFormularElement.eigenschaft  
ZeigerAufFormularElement.methode()
```

index1: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

index2: ab 0
Nummer des Formular-Elementes im Formular
muss in [] kodiert sein

beim IE:



```
document.all.logischer_formular_name.logischer_formular_element_name.eigenschaft
document.all.logischer_formular_name.logischer_formular_element_name.methode()
```

```
var ZeigerAufFormular = document.forms[index];
ZeigerAufFormular.logischer_formular_element_name.eigenschaft
ZeigerAufFormular.logischer_formular_element_name.methode()
```

index: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

```
var ZeigerAufFormular = document.forms[index1];
ZeigerAufFormular.elements[inde2].eigenschaft
ZeigerAufFormular.elements[inde2].methode()
```

index1: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

index2: ab 0
Nummer des Formular-Elementes im Dokument
muss in [] kodiert sein

```
var ZeigerAufFormular = document.forms[index1];
var ZeigerAufFormularElement = ZeigerAufFormular.elements[index2];
ZeigerAufFormularElement.eigenschaft
ZeigerAufFormularElement.methode()
```

index1: ab 0
Nummer des Formulars im Dokument
muss in [] kodiert sein

index2: ab 0
Nummer des Formular-Elementes im Dokument
muss in [] kodiert sein

Ereignisse (ausgewählte):

onreset ausgeführt beim Rücksetzen des Formulars
onsubmit ausgeführt vor dem Senden des Formulars, also verwendbar für
Formularprüfung oder Unterbinden des Sendens wegen
fehlerhaften Formulardaten

Eigenschaften zum Internet Explorer:

.acceptCharset Liste von UTF-8 Zeichen für Encoden der Eingabedaten durch den Server
Liste deklariert alle Zeichen, die nicht im Charset des Dokumentes erfasst werden
Liste wird vom Server benötigt
wenn nicht kodiert, so nur der Charset des Dokumentes verwendet
UTF-8 Zeichen: Teil des Unicode (0 bis 255)

.action Url des Servers und des dortigen Dokumentes bei Formular

Beispiele für mailto-Formen:

Standard	mailto:aaa@bbb
carbon copy	mailto:aaa@bbb?cc=mailto:ccc@ddd (mit Kopie an anderen Empfänger)
blind copy	mailto:aaa@bbb?bcc=mailto:ccc@ddd (mit Blind-Kopie an anderen Empfänger)
carbon copy und Betreff	mailto:aaa@bbb?cc=mailto:ccc@ddd&subject=betreff_text
carbon copy und Betreff und Mailtext	mailto:aaa@bbb?cc=mailto:ccc@ddd&subject=betreff_text&body=mail_text

Betreff/Mailtext analog für Standard und blind copy

Beispiel für Spam-Schutz einer Email-Adresse (Schutz vor Missbrauch nach dem Scannen der Email-Adresse durch Suchmaschine):

// test@test.de wird per Script und nicht als HTML im Quellcode kodiert

```
var user_name="test";
var host_name="test.de";
```

```
document.write( '<A HREF="mailto:'
+ user_name
+ '@'
+ host_name
+ '">'
+ user_name
```



	+ '@'
	+ host_name
	+ ''
);
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.autocomplete	Status des Autovervollständigung zum Formular
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.blockDirection	Umfluss um ein Objekt
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.enctype	Multipurpose Internet Mail Extensions (MIME) des Formulars für Encoding nach dem Senden der Daten ab IE 6.x
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.method	Methode des Senden/Empfagen der Daten an/von den Server bei Formular
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)



.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.target	Name des Ziel-Fenster bzw. Ziel-Frame
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden zum Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag



	(falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME (analog zum Attribut NAME) liefern Achtung: Objekte, die kein NAME besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per Tag-Name siehe Methode .getElementsByTagName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)



.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.reset()	löst für Formular das Event onreset aus, dessen Eventhandler die Reset-Aktion beinhaltet, die gestartet wird VOR dem Reset der Elemente (Browser setzt zurück) Eventhandler muss liefern: return true, für Ausführung des Reset durch Browser



	return false;	für Nicht-Ausführung des Reset durch Browser
Beispiel:	<pre><HTML> <HEAD> <SCRIPT> function EventHandler_AktionVorReset() { // ein Hinweis im Textarea anzeigen ID_Textarea.value += "Formular zuruecksetzen ????"; // wirklich rücksetzen ??? return(confirm("Wirklich rücksetzen ?")); // true so Reset durch Browser ausführen lassen // false so kein Reset durch Browser } </SCRIPT> </HEAD> <BODY> <DIV> <FORM NAME="Formular" onreset="EventHandler_AktionVorReset();"> <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
 <BUTTON onclick="form.reset();">OnReset ausloesen</BUTTON>

 <INPUT TYPE="reset" VALUE="oder hiermit zuruecksetzen" onclick="form.reset()" > </FORM> </DIV> <TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA> </BODY> </HTML></pre>	
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird	
.setActive()	Objekt muss an sich schon renderbar sein Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen	
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert	
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.	
.setExpression()	ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert	
.submit()	löst für Formular das Event onsubmit aus, dessen Eventhandler die Submit-Aktion beinhaltet, die gestartet wird VOR dem Senden der Elemente (Browser sendet) Eventhandler muss liefern: return true, für Ausführung des Submit durch Browser return false; für Nicht-Ausführung des Submit durch Browser	

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function EventHandler_AktionVorSubmit()
{
    // ein Hinweis im Textarea anzeigen
    ID_Textarea.value += "Formular senden ????";}

    // wirklich senden ???
    return( confirm("Wirklich senden ?"));
    // true so Submit durch Browser ausführen lassen
    // false so kein Submit durch Browser
}
</SCRIPT>
</HEAD>
<BODY>
<DIV>
<FORM NAME="Formular" ACTION=" ..." METHOD=" "
    onsubmit="EventHandler_AktionVorSubmit();">

```



```

        <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
        <BR>
        <BUTTON onclick="form.submit();">OnSubmit auslösen</BUTTON>
        <BR><BR>
        <INPUT TYPE="submit" VALUE="oder hiermit senden"
            onclick="form.submit()"
        >
    </FORM>
</DIV>
<TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>

```

.swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.14.1. Objekt document.form.input und seine Varianten beim Internet Explorer

Formularinhalte können diverse HTML-Elemente sein., vorallem Elemente, die Daten speichern können wie z.B. TEXTAREA, SELECT, OPTION etc.. Ein sehr beliebtes HTML-Element ist das input Objekt, das in diversen Varianten kodiert werden kann. **Jedes Element** innerhalb

von <FORM> .. </FORM> wird zum Kind des Objektes form. Daher ist Punktnotation nötig, um den Formularinhalt, also die Kinder des Objektes form, referenzieren zu können. Der Objekttyp input.image kann **nicht für eine Formular** referenziert werden, wenn er innerhalb <FORM> ... </FORM> kodiert wurde. Für input.image ist immer die children Collection zu verwenden.

Nachfolgend werden die Varianten des input Objektes kurz beschrieben, wobei nur eine Auswahl von Eigenschaften und Methoden genannt wird. Aus Übersichtsründen sind weitere Eigenschaften und Methoden direkt bei der Beschreibung vom input Objekt **und nicht hier** zu finden. Ereignisse werden alle beim Objekt event beschrieben. Die Variante **input.image** wird nicht beschrieben.

Die Varianten des input Objekt basieren z.T. auch auf anderen Objekten.

Im Objekt input muss das Attribut TYPE mit dem Objekttyp belegt werden, von dem die Variante des Objektes input abstammt. Der Wert vom Attribut TYPE kann wahlweise mit " " bzw. ' ' oder ohne " " bzw. ' ' kodiert werden, wobei Gross-Kleinschreibung egal ist. Der Objekttyp image kann **nicht für eine Formular** referenziert werden, wenn er innerhalb <FORM> ... </FORM> kodiert wurde.

Die Varianten von input haben nur Sinn, wenn **zugleich** Eventhandler kodiert werden, da ein Formular die User-Interaktion behandeln sollte.

4.3.2.2.4.3.14.1.1. Objekt document.form.input.button des Internet Explorer

Diese Objekt basiert zusätzlich auf dem button Objekt (siehe dort).

Erzeugung durch HTML:

Beispiel:

```

<INPUT ID="ID_Input"
    TYPE=button
    VALUE="button_beschriftung"
    NAME="logischer_formular_element_name"
    onblur="eventhandler1"
    onfocus="eventhandler2"
    onclick="eventhandler3"
    onmousedown="eventhandler4"
    onmouseup="eventhandler5"
>

```

Beispiel:

```

<HEAD>
<SCRIPT>
    function ValueAendern()
    { ID_Input1.value = "Neuer Wert von VALUE";}

    function FarbeAendern()
    { ID_Input2.style.backgroundColor = "aqua";}

    function Anzeige()
    {alert(event.propertyName + " wurde verändert");}

</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE=button ID="ID_Input1"
        VALUE="Click um VALUE zu ändern"
        onclick="ValueAendern()"
        onpropertychange="Anzeige()"
    >
    <INPUT TYPE=button ID="ID_Input2"
        VALUE="Click um Farbe zu ändern"
        onclick="FarbeAendern()"
    >

```




```
onpropertychange="Anzeige()"
>
```

```
</BODY>
```

Eigenschaften (ausgewählte) :

.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	liefert immer den Typ button
.value	entspricht VALUE

Methoden (ausgewählte):

.blur()	entfernt den Cursor vom Button
.click()	bewirkt einen Klick auf die Schaltfläche belegt CHECKED und .checked und .defaultChecked
.focus()	setzt den den Cursor auf das Button

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn User der Cursor vom Button entfernt
onclick	ausgelöst, wenn User auf das Button klickt
onfocus	ausgelöst, wenn User den Cursor auf das Button bewegt
onmousedown	ausgelöst, wenn User auf dem Button die Maustaste niederdrückt
onmouseup	ausgelöst, wenn User auf dem Button die gedrückte Maustaste loslässt

4.3.2.2.4.3.14.1.2. Objekt document.form.input.checkbox (Abhak-Kästchen) des Internet Explorer**Erzeugung durch HTML:**

Beispiel:

```
<INPUT ID="ID_Input"
      TYPE=checkbox
      VALUE="wert_der_gesendet_wird" // Standardwert ist "on"
      NAME="logischer_formular_element_name"
      CHECKED // Checkbox ist mit Erstellung bereits markiert
      onblur="eventhandler1"
      onfocus="eventhandler2"
      onclick="eventhandler3"
>
```

Beispiele:

Beispiel 1:

```
<HEAD>
<SCRIPT>
    function Anzeige()
    {alert("Checkbox-Status = " + ID_Checkbox.checked);}
</SCRIPT>
</HEAD>
<BODY>
    selektiere !
    <INPUT TYPE=checkbox
          ID="ID_Checkbox"
          NAME="checkbox1"
          onclick=Anzeige()
    >
</BODY>
```

Beispiel 2:

```
<INPUT TYPE=checkbox
      ID="ID_InputCheckbox"
      CHECKED
      DISABLED
>
<SPAN STYLE="font-weight:bold"
      onclick="ID_InputCheckbox.status=false"
>
    ablehnen
</SPAN>
<SPAN STYLE="font-weight:bold"
      onclick="ID_InputCheckbox.status=true"
>
    annehmen
</SPAN>
```

Beispiel 3:

```
<HTML>
<HEAD>
<SCRIPT>
    function ClickMitFocus()
    {
        ID_CheckBox.focus();
        ClickOhneFocus();
    }
</SCRIPT>
```



```

    }

    function ClickOhneFocus ()
    { ID_CheckBox.click();}
</SCRIPT>
<SCRIPT FOR= ID_CheckBox EVENT=onfocus>
    alert("Check Box hat Focus");
</SCRIPT>
</HEAD>
<BODY>
    <INPUT Type="CHECKBOX" ID="ID_CheckBox"></INPUT>
    <BR>
    <BUTTON onclick="ClickMitFocus()">Klick mit Fokus</BUTTON>
    <BR>
    <BUTTON onclick="ClickOhneFocus()">Klick ohne Fokus</BUTTON>
</BODY>
</HTML>

```

Eigenschaften (ausgewählte):

.checked	nur lesen ist true, wenn Checkbox abgehakt
.defaultChecked	ist true, wenn CHECKED in <INPUT> kodiert kann belegt werden mit true oder false false bewirkt CHECKED in <INPUT> wird unwirksam
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	liefert immer Typ checkbox
.value	entspricht VALUE

Methoden (ausgewählte):

.blur()	entfernt Cursor von der Checkbox
.click()	bewirkt markieren oder nicht markieren belegt CHECKED und .checked und .defaultChecked
.focus()	setzt Cursor auf die Checkbox

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn User der Cursor vom Checkbox entfernt
onclick	ausgelöst, wenn User auf die Checkbox klickt
onfocus	ausgelöst, wenn User den Cursor auf die Checkbox bewegt

Hinweise: onclick für weitere Reaktionen aufgrund markieren bzw. nicht markieren

4.3.2.2.4.3.14.1.3. Objekt document.form.input.fileupload des Internet Explorer**Erzeugung unter HTML:**

Beispiel:

```

<INPUT ID="ID_Input"
    TYPE=file
    NAME="logischer_formular_element_name"
    onblur="eventhandler1"
    onchange="eventhandler2"
    onfocus="eventhandler3"
>

```

Eigenschaften (ausgewählte):

.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	enthält den Dateinamen, nur lesen

Methoden (ausgewählte):

.blur()	entfernt Cursor vom Eingabefeld
.focus()	setzt den Cursor auf das Eingabefeld
.select()	markiert den Inhalt des Eingabefeldes, auf dem per .focus() der Cursor gesetzt sein muss

Events (ausgewählte):

onblur	ausgelöst, wenn User der Cursor aus der Eingabezeile bewegt
onchange	ausgelöst, wenn User auf den Inhalt der Eingabezeile ändert
onfocus	ausgelöst, wenn User den Cursor auf die Eingabezeile bewegt

4.3.2.2.4.3.14.1.4. Objekt document.form.input.hidden des Internet Explorer**Erzeugung unter HTML:**

Beispiel:

```

<INPUT ID="ID_Input"
    TYPE=hidden
    NAME="logischer_formular_element_name"
    VALUE="wert"
>

```

Beispiel:

```

<FORM ACTION="mailto:test@test.de" METHOD=GET>

```



```

<INPUT NAME=subject TYPE=hidden
      VALUE="Testt%20Product%20Information%20Anforderung"
>
      volle Mailadresse eingeben
<BR>
<TEXTAREA NAME=body COLS=40></TEXTAREA>
<INPUT TYPE=submit VALUE="absenden "
</FORM>

```

Eigenschaften (ausgewählte):

.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	entspricht VALUE

4.3.2.2.4.3.14.1.5. Objekt document.form.input.password des Internet Explorer**Erzeugung unter HTML:**

Beispiel:

```

<INPUT ID="ID_Input"
      TYPE=password
      NAME="logischer_formular_element_name"
      VALUE="vorbelegung"
      SIZE="breite_eingabe_feld_in_zeichen"
      onblur="eventhandler1"
      onchange="eventhandler2"
      onfocus="eventhandler3"
      onselect="eventhandler4"
>

```

Beispiel:

```

<INPUT TYPE="password" AUTOCOMPLETE="off">

```

Eigenschaften (ausgewählte):

.defaultValue	ist standardgemäß eine Leerkette kann verändert werden
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	entspricht VALUE enthält die Benutzereingabe

Methoden (ausgewählte):

.blur()	entfernt Cursor vom Eingabefeld
.focus()	setzt Cursor auf Eingabefeld
.select()	markiert den Inhalt des Eingabefeldes, da zuvor mit focus() den Cursor bekommt

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn User der Cursor aus dem Eingabefeld bewegt
onfocus	ausgelöst, wenn User den Cursor auf das Eingabefeld bewegt

4.3.2.2.4.3.14.1.6. Objekt document.form.input.radio des Internet Explorer**Erzeugung unter HTML:**

Beispiel:

```

<INPUT ID="ID_Input"
      TYPE=radio
      VALUE="radio_wert_der_gesendet_wird_wenn_button_markiert_ist"
      NAME="logischer_radio_name" bzw. "logischer_radio_gruppen_name"
                                     // Radiobutton mit identischem Namen gehören einer Gruppe an
                                     // in der Gruppe kann pro Zeitpunkt maximal nur 1
                                     // Button markiert sein

      CHECKED
      onblur="eventhandler1"
      onfocus="eventhandler2"
      onclick="eventhandler3"
>

```

Beispiele:

Beispiel 1:

```

<HEAD>
<SCRIPT>
      function KanalVerteilung(Wert)
      { ID_bgsound.balance = Wert; }

      function GenauEinmal()
      {
          ID_bgsound.loop = 1;
          ID_bgsound.src = ID_bgsound.src; // restart
      }

```



```

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>
    <BUTTON onclick="Endlos()"></BUTTON>
    <BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung(10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung (0)"></BUTTON>
    <B>Volume control:</B>&nbsp;
    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-10;"
    >Mute

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-7;"
    >25% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
        onpropertychange="ID_bgsound.volume=-5;"
    >50% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
    >75% Volume

    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
    >100% Volume
</BODY>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Erzeuge()
    {
        var Zeiger = document.createElement(
            "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Eins'>"
        );
        document.body.insertBefore(Zeiger);

        Zeiger = document.createElement(
            "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Zwei'>"
        );
        document.body.insertBefore(Zeiger);
    }
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE="BUTTON"
        ONCLICK=" Erzeuge()"
        VALUE="Zwei Radio Buttons erzeugen">

    <BR>
    <INPUT TYPE="BUTTON"
        ONCLICK="alert(document.body.outerHTML)"
        VALUE="Click um HTML zu sehen">

    <BODY>
</HTML>

```

Eigenschaften (ausgewählte):

.checked	ist true wenn Radiobutton markiert ist (sonst false)
.defaultChecked	ist true wenn CHECKED in INPUT kodiert ist
	kann neu gesetzt werden --> Wirkung von CHECKED aus INPUT aufgehoben
.form	Zeiger auf das Formular (Formular als Container)
	ab IE 6.x für Elemente fieldSet, label, legend
.index	Index des ausgeählten Elementes
.length	Anzahl Radiobutton innerhalb der Gruppe
.name	entspricht NAME
.type	entspricht TYPE



.value	entspricht VALUE
Methoden (ausgewählte):	
.blur()	entfernt Cursor vom Radiobutton
.click()	bewirkt markieren bzw. entmarkieren des Buttons (Click auf das Radiobutton) sowie setzen der Eigenschaft checked auf true für das angeklickte Button Eigenschaft checked auf false für alle anderen Button der Gruppe
.focus()	setzt den Cursor auf das Radiobutton
Ereignisse (ausgewählte):	
onblur	ausgelöst, wenn User den Cursor vom Radiobutton entfernt
onclick:	ausgelöst, wenn User auf das Radiobutton klickt Eventhandler für weitere Reaktion nach Anklicken: muss return true; oder return false; besitzen: true, so erfolgt das Senden des radio_wertes auch wirklich false, so erfolgt kein Senden trotz angeklicktem Button
onfocus	ausgelöst, wenn User den Cursor auf das Radiobutton bewegt

4.3.2.2.4.3.14.1.7. Objekt document.form.input.reset des Internet Explorer

Erzeugung unter HTML:

Beispiel:

```
<INPUT ID="ID_Input"
      TYPE=reset
      VALUE="reset_button_beschriftung"
      NAME="logischer_formular_element_name"
      onblur="eventhandler1"
      onfocus="eventhandler2"
      onclick="eventhandler3"
      onreset="eventhandler4"
>
```

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function EventHandler_AktionVorReset()
    {
        // ein Hinweis im Textarea anzeigen
        ID_Textarea.value += "Formular zuruecksetzen ???";

        // wirklich rücksetzen ???
        return( confirm("Wirklich rücksetzen ?"));
        // true so Reset durch Browser ausführen lassen
        // false so kein Reset durch Browser
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV>
        <FORM NAME="Formular" onreset="EventHandler_AktionVorReset();">
            <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
            <BR>
            <BUTTON onclick="form.reset();">OnReset ausloesen</BUTTON>
            <BR><BR>
            <INPUT TYPE="reset" VALUE="oder hiermit zuruecksetzen" onclick="form.reset()">
        </FORM>
    </DIV>
    <TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>
```

Eigenschaften (ausgewählte):

.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	entspricht VALUE wenn VALUE nicht kodiert, so "Reset" als Standard nur lesbar

Methoden (ausgewählte):

.blur()	entfernt den Cursor vom Resetbutton
.click()	bewirkt Anklicken des Resetbuttons und löschen des Formulars
.focus()	setzt den Cursor auf das Resetbutton

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn der User den Cursor vom Resetbutton entfernt
onclick	ausgelöst, wenn der User auf das Resetbutton clickt
onfocus	ausgelöst, wenn User den Cursor auf das Resetbutton bewegt
onreset	ausgelöst direkt vor dem Formular-Reset durch den Browser



Eventhandler für weitere Reaktionen aufgrund des Anklickens auslösen:
 muss return true; oder return false; liefern
 true, so Reset auch wirklich ausgeführt
 false, so Reset NICHT ausgeführt trotz Anklicken

4.3.2.2.4.3.14.1.8. Objekt *document.form.input.submit* des Internet Explorer

Erzeugung unter HTML:

Beispiel:

```
<INPUT ID="ID_Input"
      TYPE=submit
      NAME="logischer_formular_element_name"
      VALUE="beschriftung_des_button"
      onblur="eventhandler1"
      onfocus="eventhandler2"
      onclick="eventhandler3"
      onsubmit="eventhandler4"
>
```

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function EventHandler_AktionVorSubmit()
{
    // ein Hinweis im Textarea anzeigen
    ID_Textarea.value += "Formular senden ???";

    // wirklich senden???
    return( confirm("Wirklich senden ?"));
    // true so Submit durch Browser ausführen lassen
    // false so kein Submit durch Browser
}
</SCRIPT>
</HEAD>
<BODY>
<DIV>
<FORM NAME="Formular" ACTION=" ..." METHOD=" "
      onsubmit="EventHandler_AktionVorSubmit();">
  <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
  <BR>
  <BUTTON onclick="form.submit();">OnSubmit auslösen</BUTTON>
  <BR><BR>
  <INPUT TYPE="submit" VALUE="oder hiermit senden" onclick="form.submit();">
</FORM>
</DIV>
<TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>
```

Eigenschaften (ausgewählte):

.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	entspricht VALUE wenn VALUE in <INPUT> nicht kodiert, so "Submit Query" als Wert nur lesen

Methoden (ausgewählte):

.blur()	entfernt den Cursor vom Submitbutton
.click()	bewirkt standardgemäß sofortiges Abschießen des Formulares (aufgrund Anklicken des Buttons), wenn es sich NICHT um E-Mail handelt --> bei E-Mail erfolgt erst Nachfrage
.focus()	setzt den Cursor auf das Submitbutton

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn User den Cursor vom Submitbutton entfernt
onclick	ausgelöst, wenn User das Submitbutton clickt
onfocus	ausgelöst, wenn User den Cursor auf das Submitbutton bewegt
onsubmit	ausgelöst direkt vor dem Formular-Senden durch den Browser Eventhandler für weitere Reaktionen aufgrund des Anklickens auslösen: muss return true; oder return false; liefern true, so Senden auch wirklich ausgeführt false, so Senden NICHT ausgeführt trotz Anklicken

4.3.2.2.4.3.14.1.9. Objekt *document.form.input.text* des Internet Explorer

Dieses Objekt basiert zusätzlich auf dem Objekt text (siehe dort).

Erzeugung unter HTML:



Beispiel:

```
<INPUT ID="ID_Input"
      TYPE=text
      NAME="logischer_formular_element_name"
      VALUE="zeichenkette_als_vorgabe_wert"
      SIZE="breite_eingabefeld_in_zeichen"
      onblur="eventhandler1"
      onchange="eventhandler2"
      onfocus="eventhandler3"
      onkeydown="eventhandler4"
      onkeypress="eventhandler5"
      onkeyup="eventhandler6"
      onselect="eventhandler7"

>
```

Beispiele:

Beispiel 1:

```
<LABEL FOR="ID_Input" ACCESSKEY="I">
    #<SPAN>1</SPAN>:
    Alt+I fuer Sprung zur Textbox
</LABEL>
<INPUT TYPE="text"
      ID="ID_Input "
      NAME="T1"
      VALUE=text1
      SIZE="20"
      TABINDEX="1"

>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
    function Antworten(Wert)
    {Wert ? alert("Ja") : alert("Nein");}
</SCRIPT>
</HEAD>
<BODY>
    <P>
        <INPUT type="text" ID="ID_Input" VALUE="Test">
        <BUTTON onclick="Antworten(ID_Input.isMultiLine);">
            Mehrzeiligkeit eines INPUT TYPE=text
        </BUTTON>
    </P>
    <P>
        TEXTAREA:
        <TEXTAREA ID="ID_Textarea">
            Test
        </TEXTAREA>
        <BUTTON onclick="Antworten(ID_Textarea.isMultiLine);">
            Mehrzeiligkeit eines Textbereiches
        </BUTTON>
    </P>
</BODY>
</HTML>
```

Beispiel 3:

```
<INPUT TYPE=text SIZE=33>
```

Beispiel 4 für Auslesen eines Eingabefeldes:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function gruss()
    {alert("Hallo " + document.meinFormular.Eingabe.value + "!");}
// -->
</SCRIPT>
</HEAD>
<BODY>
    Bitte Namen eingeben und danach den Knopf drücken
    <FORM NAME="meinFormular">
        <INPUT TYPE="text"
            NAME="Eingabe"
```



```

        VALUE=""
    >
    <BR>
    <INPUT TYPE="button"
        NAME="Knopf"
        VALUE="Bitte drücken"
        onClick="gruss()"
    >
</FORM>
</BODY>
</HTML>

```

Beispiel 5 für Wert einem Eingabefeld zuweisen:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function umrechnung(celsius)
    {
        var fahrenheit;
        fahrenheit = 9 / 5 * celsius + 32;
        document.Formular.Ausgabe.value = fahrenheit;
    }
// -->
</SCRIPT>
</HEAD>

<BODY>
Bitte geben Sie einen Temperaturwert in Celsius ein:
<FORM NAME="Formular">
    <INPUT TYPE="text"
        NAME="Eingabe"
        VALUE=""
    >
    <BR>
    <INPUT TYPE="button"
        NAME="Knopf"
        VALUE="Berechnung"
        onClick="umrechnung(this.form.Eingabe.value)"
    >
    <BR>
Entpricht
    <INPUT TYPE="text"
        NAME="Ausgabe"
        VALUE=""
    >
    Fahrenheit
</FORM>
</BODY>
</HTML>

```

Beispiel 6 für Laufschrift in einem Formular

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    text="Laufschrift";

    function anzeigen()
    {
        teilkette=text.substring(0, 1);
        text=text.substring(1, text.length);
        text= text + teilkette;

        window.document.formular.schrift.value=text;
        setTimeout('anzeigen()',200);
    }
//-->
</SCRIPT>
</HEAD>

<BODY>

```




```

<FORM NAME="formular">
  <INPUT TYPE=button
    VALUE="Laufschrift im Formularfeld"
    onClick="anzeigen()"
  >
  <INPUT TYPE="text"
    NAME="schrift"
    SIZE="20"
    READONLY
  >
</FORM>
</BODY>
</HTML>

```

Eigenschaften (ausgewählte):

.defaultValue	entspricht VALUE
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.name	entspricht NAME
.type	entspricht TYPE
.value	entspricht nicht VALUE enthält den eingegebenen Text

Methoden (ausgewählte):

.blur()	entfernt den Cursor aus dem Eingabefeld (deaktiviert Textfeld für Eingabe)
.focus()	setzt den Cursor auf das Eingabefeld
.select()	markiert den Text im Eingabefeld Cursor muss zuvor mit focus() auf das Eingabefeld bewegt worden sein

Ereignisse (ausgewählte):

onblur	ausgelöst, wenn User den Cursor vom Eingabefeld entfernt
onchange	ausgelöst, wenn User das Eingabefeld inhaltlich ändert
onfocus	ausgelöst, wenn User den Cursor auf das Eingabefeld bewegt
onselect	ausgelöst, wenn Teil des Inhaltes vom Eingabefeld markiert wird

4.3.2.2.4.3.14.2. document.form.elements Collection des Internet Explorer

Felder der Zeiger auf alle Formularkomponenten **außer** Objekt input.image, das damit **im** Formular nicht referenzierbar wird:

Für input image ist immer die children Collection zu verwenden.

Reihenfolge der Feldelemente laut der Kodierung der Elemente im Formular

Syntax:

```

[ var ZeigerAufFeld = ] zeiger_auf_form_objekt.elements
[ var ZeigerAufFeldElement = ] zeiger_auf_form_objekt.elements[Index, [SubIndex]]

```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

zeiger_auf_form_objekt z.B. laut ID-Attribut

Eigenschaften:

.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
---------	--

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.tags()	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.15. document.forms Collection des Internet Explorer

Feld der Zeiger aller Formular-Objekte im Dokument

Elementefolge laut HTML-Coding

Syntax:

```

[ var ZeigerAufFeld = ] document.forms
[ var ZeigerAufFeldElement = ] document.forms[Index, [SubIndex]]

```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist



Integer als Unterindex also Unterelement eines Elementes

Eigenschaften:**.length**

Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:**.item()**

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem()

Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.tags()

Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns()

Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.16. frame Objekt

Hinweis: Wenn ein Fenster mit dem Unterstrichsymbol rechts oben in der Fensterecke minimiert wurde, dann ist es durch Windows in die Hintergrund-Prioritätenfolge eingeordnet worden und arbeitet im Hintergrund bzw. garnicht. Mit anschliessendem Maximieren (Symbol in der Fensterleiste rechts oben) wird das Fenster wieder angezeigt und das geladene Dokument **erneut** geöffnet. Konsequenz daraus ist, dass dieses Maximieren einem Neustart des Dokumentes entspricht, auch wenn der Programmierer diesen Zustand nicht erwünscht. Sound, der z.B. mit Start des Dokumentes erzeugt wird, erklingt erneut mit Maximierung nach einer Fensterminimierung. Analogon ist die Fenstereinengung im Browser durch z.B. Ein- und Ausblenden der Favoritenleiste. Dieses Verhalten des Fensters mit seinem Dokument ist aber **nicht identisch** mit dem Verhalten nach einer Fenstergrößenänderung z.B. durch Rahmenverschiebung (resize). Die Aktionen des Fensters und seines Dokumentes bezüglich Resize müssen programmiert werden, sind also nicht standardmäßig vorhanden - im Gegensatz zum Verhalten mit/nach Fensterminimierung/-maximierung per Symbole in der rechten oberen Ecke der Fensterleiste. Sollte ein Frameset minimiert werden, dann wird das für den gesamten Frameset getan (Frameset hat 1 Fenster für alle Frames gesamt), allerdings mit Maximierung wird auch das Frameset-Dokument neu geladen. Wichtig dabei sind für Zeiger-Bezüge der Frames auf den Frameset per parent-Zeiger, dass die Daten im Frameset-Dokument mit Maximierung initialisiert werden und somit ebenfalls Daten der Frames, die im Frameset-Dokument abgelegt wurden, also z.B. Daten, die Verhaltensweisen der Frames vor einer Änderung der Frames-Inhalte gespeichert haben.

Erzeugung unter HTML:

```
<FRAMESET
    ROWS="zeilen_liste" oder COLS="spalten_liste" // Zeilenliste=Liste der Framehöhen
                                                    // mit Kommatrennung
                                                    // Spaltenliste=Liste der Framebreiten
                                                    // mit Kommatrennung
    onLoad="evenhandler1" // Anweisungen aktiviert NACH laden ALLER Frames
    onUnload="eventhandler2"
    onerror="eventhandler3"
    onBlur="eventhandler4"
    onFocus="eventhandler5"
>

[<FRAME
    SRC="frame_url" // url der HTML-Seite als Frameinhalt
    NAME="logischer_frame_name"
>]
.....
</FRAME>
.....
</FRAMESET>
```

Rahmen zwischen den einzelnen Fenstern des Framesets entfernen:

```
<FRAMESET    BORDER="0"
              FRAMEBORDER="0"
              FRAMESPACING="0"
              .....
>
```

Hinweise:

BORDER	bei Netscape Breite des Rahmens >= 0 Pixel	
FRAMEBORDER	bei IE Rahmenanzeige 0 oder "no" 1 oder "yes"	aus ein
FRAMESPACING	bei IE Rahmenbreite >=0 Pixel	

Zugriff:

document.ID_Frame.eigenschaft



```
document.ID_Frame.methode()
```

```
document.frames[index].eigenschaft
document.frames[index].methode()
```

```
index:      ab 0
            Nummer des Frames im Dokument
            muss in [ ] kodiert sein
```

Möglichkeiten des Laden eines Dokumentes:

Laden eines fremden Dokumentes innerhalb eines Frame ist rechtlich nicht zulässig (Abmahnungsgefahr), wenn der Eigentümer der Fremdseite nicht explizit zugestimmt hat.

Laden eines fremden Dokumentes ohne Framedarstellung:

```
<HTML>
  <HEAD>
    <SCRIPT LANGUAGE="JavaScript">
      <!--
        function laden()
        { location.href = "http://www.test.de";}
      // -->
    </SCRIPT>
  </HEAD>

  <BODY>
    <FORM>
      <INPUT TYPE="button"
        VALUE="www.test.de anwählen "
        onclick="laden()">
    </FORM>
  </BODY>
</HTML>
```

Eigenes Dokument wird durch fremde Webseite geladen:

CopyRight-Meldung auf fremden Host erzeugen:

Beispiel 1:

```
// In diesem Beispiel schreibt das Script Text auf die "entführte" Seite!
var HostUrl="www.test.de";

if (      (parent !=null)
    &&    (parent != self)
    )
{
    var host=parent.location.hostname;

    if(host != HostUrl)
    {
        document.write(
            "Diese Seite wurde ausgeliehen bei "
            + "<A HREF='" + location.href
            + "' TARGET='_parent'"
            + ">"
            + HostUrl
            + "</A>"
        );
    }
}
```

Beispiel 2:

```
<BODY>
.....
if (      (parent != null)
    &&    (parent != self)
    )
{
    var  meine_url = "www.test.de"
    var  mein_host_name = "http://" + meine_url;

    var  fremder_host_name = parent.location.hostname;
```



```

        if ( fremder_host_name != mein_host_name)
        {
            document.write(      " Diese Seite liegt auf "
                                + "<A HREF=\"" + location.href + "\"\"
                                + " TARGET=\"_parent\"\"
                                + ">"
                                + "</A>"
                                + " und stammt von "
                                + mein_host_name
                                );
        }
    }
    .....
</BODY>

```

Framedarstellung der eigenen Webseite sofort und ohne Bildschirmmeldung aktivieren:

Dieses Coding muss in jedes HTML-Dokument, das in einem Frame geladen wird. Bei Einzelaufwurf des Dokumentes wird automatisch die Seite mit dem FRAMESET aktiviert, also die vollständige Framedarstellung.

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
    var EigenerHost="http://www.test.de";
        // window.location.hostname ist Leerkette, wenn Browser offline
    var StartSeite="_start.html";

    function InaktiveFrameDarstellungAktivieren()
    {
        if (top.frames.length == 0)
        {
            // keine Frame-Darstellung aktiv, also diese aktivieren
            top.location.href = StartSeite; // muss das FRAMESET enthalten !
        }
    }

    if (    (parent != null) // Elternobjekt existiert
        && (parent != self) // Eltern sind vorhanden Eltern: dieses Dokument (self) ist ein Kind
        )
    {
        // aktuellen ElternHost ermitteln
        var ElternHost=parent.location.hostname;

        // ElternHost prüfen ob eigener und nicht leer, also Browser online ist
        if (    (ElternHost != "")                                // Browser ist online
            && (ElternHost != EigenerHost)
            )
        {
            // fremder Host, also als oberstes Fenster nun die Startseite anzeigen
            // und damit den eigenen Host aktivieren
            top.location.href=EigenerHost + "/" + StartSeite;
        }
        else
        {
            // eigener Host und/oder Browser ist offline
            InaktiveFrameDarstellungAktivieren();
        }
    }
    else
    {
        // Elternobjekt existiert nicht und/oder dieses Dokument (self) ist kein Kind
        InaktiveFrameDarstellungAktivieren();
    }
}
-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Reload eines Dokumentes mit allen seinen FRAMES:



```
function frame_neu_laden()
{
    for (var i=0; i<windows.FRAMES.length; i++)
    { windows.frames[i].location.reload(false); }
}

<FRAMESET onResize="frame_neu_laden()>
```

Datei ohne FRAMESET in eine Datei mit FRAMESET laden

```
if (self.location == top.location)
{ location.href="/FRAMESET.html?" + escape(location.pathname); }
```

Mehrere Frameinhalte gleichzeitig ändern:**Inhalte zweier Frames tauschen:**

Kodierung im Dokument, dass die beiden Frames definiert !

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
                          logischer_name_rahmen2,html_datei2
                          )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch(0, 'a.html', 1,b.html)">tauschen</A>
```

Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framenamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument. Argumententrenner sind Doppelpunkte.

Tausch:

```
erster Frame retten und dann mit zweiten Frame überschreiben
zweiten Frame mit drittem Frame überschreiben
.....
vorletzten Frame mit letztem Frame überschreiben
letzten Frame mit geretteten ersten Frame überschreiben

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)          // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);

                for(var i = 0; i < arguments.length; i++)    // Index ab 0
                {
                    pos_doppelpunkt = arguments[i].indexOf(":");

                    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
                    {
                        // ersten zu ersetzenden Framenamen retten

                        if (i=0)
                        { var rette_logischer_framename =
                          arguments[i].substring(0, pos_doppelpunkt)
                        }

                        // tauschen der logischen Framenamen
                        logischer_framename_zu_ersetzender_frame =
                          arguments[i].substring(0, pos_doppelpunkt);
```



```

logischer_framename_ersetzender_frame =
    arguments[i].substring(0, pos_doppelpunkt + 1);

frames[logischer_framename_zu_ersetzender_frame].location.href =
    logischer_framename_ersetzender_frame;
}

frames[logischer_framename_ersetzender_frame].location.href=
    rette_logischer_framename;
}

}

// -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>

```

Frame und Datenaustausch:

Zwischen Frames können Daten ausgetauscht werden.

Beispiel Adressierung eines Frame über das FRAMESET

```

<FRAMESET .... >
    <FRAMESET ..... >
        <FRAME SRC="test.html" NAME="test">
        <FRAME SRC="test1.html" NAME="test1">
    </FRAMESET>
    <FRAME SRC="test2.html" NAME="test2">
</FRAMESET>

parent.test2.location.href="neu.html";    // Laden von neu.html in den Frame test2

```

Beispiel: Auf Objekte im FRAMESET-Dokument durch ein FRAME-Dokument zugreifen

```

im FRAMESET-Dokument wird kodiert      var Kette="Hallo !";

im FRAME-Dokument wird auf Kette zugegriffen:  alert(parent.Kette);

```

Beispiel: Auf Objekte im FRAME-Dokument durch das FRAMESET-Dokument zugreifen

```

im FRAME-Dokument wird kodiert      var Kette="Hallo !";
                                     <FRAME ...NAME ="test2" ...>

im FRAMESET-Dokument wird auf Kette zugegriffen:      alert(parent.test2.Kette);

```

Beispiel: Textdaten-Übergabe durch an Url angehängte Textdaten

**quelle.htm: lädt ziel.htm
 übergibt Textdaten an ziel.htm**

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function ziel_seite_laden_mit_datenuebergabe(uebergabe_datan)
    {location.href = "ziel.htm?" + escape(uebergabe_datan);}
    // Url von ziel.htm als Suchbegriff mit angehangenen Daten merken, die per escape() in das
    //      Url-Format konvertiert wurden

// -->
</SCRIPT>
</HEAD>

<BODY>
An ziel.htm zu &uuml;bergabenden Text eingeben:
<FORM>
    <TEXTAREA       NAME=eingabe
                     ROWS=5
                     COLS=40
    >
    </TEXTAREA>
    <BR>
    <INPUT TYPE=button

```



```

        VALUE="Zielseite laden"
        onClick=" ziel_seite_laden_mit_datenuebergabe(this.form.eingabe.value)">
    </FORM>
</BODY>
</HTML>

ziel.htm:           wird durch quelle.htm geladen
                     erhält Textdaten von quelle.htm

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    datenuebernahme()
    {
        // Url mit angehangenen Daten holen
        // search liefert ?daten
        uebernahme_daten= location.search;

        // ? abschneiden
        uebernahme_daten= uebernahme_daten.substring(1, uebernahme_daten.length);

        // unescape: von Url-Format nach Zeichenkette
        document.ausgabe.ausgabefeld.value=unescape(uebernahme_daten);
    }

// -->
</SCRIPT>
</HEAD>

<BODY onLoad=" datenuebernahme ()">
Von quelle.htm &uuml;bernommener Text:
    <FORM NAME=ausgabe>
        <TEXTAREA      NAME=ausgabefeld
                        ROWS=10 COLS=40>
        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Beispiel: Textdaten-Übergabe durch Fenster-Handle

Die Textdaten und die der Javascript-Code, der die Textdaten verarbeitet, sind **getrennte** HTML-Dokumente. Nachteil dieser Variante ist, dass in beiden Dokumenten die logischen Namen vom Formular und dem Textarea identisch

kodiert werden müssen, also eine doppelte Verwaltung nötig ist.

HTML-Dokument, das die Textdaten enthält:

```

<HTML>
<BODY>
    <FORM NAME=formular>
        <TEXTAREA NAME=text_area>
            // Hier die Textdaten als Wert von TEXTAREA
        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Dokument, das die Textdaten verarbeitet:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function TextAreaWertLesen(url)
    {
        // url enthält den Pfad und den Namen des Dokumentes, dass die
        // Textarea-Werte enthält
        var handle = window.open(url_der_datendatei,"", "width=100,height=100");

        // Handle erzeugen und Fenster mit dem Textarea anzeigen
        // (Fenstergröße ist egal)
        var data = handle.document.forms[formular].elements[text_area].value;
    }

```



```

        handle.close();

        return data;
    }

    // -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

frame Objekt des Internet Explorer:

Frame muss im FRAMESET liegen.

Die Collection frames wird unter document.frames beschrieben !

Zugriff auf Frame-Eigenschaft:

Beispiel

```
var Wert = document.all.ZeigerAufFrame.eigenschaft;
```

mit ZeigerAufFrame laut ID-Attribut

```
<FRAME ID="ZeigerAufFrame" ..... >
```

Transparenter Content des Frame:

ab IE 5.5

für den Frame muss das Attribut .ALLOWTRANSPARENCY auf true gesetzt sein

Im Dokument, das in den Frame geladen wird, muss im BODY die Hintergrundfarbe (background-color oder BGCOLOR-Attribut) auf transparent gesetzt sein.

Eigenschaften:

.allowTransparency	Transparenz ein/aus Transparenz: z.B. Hintergrundfarbe des Elternobjektes wird zur Hintergrundfarbe des Frames Achtung: Wenn Frame-eigenes STYLE-Attribut mit Wert für background-color also STYLE="background-color: farb_bezeichner" kodiert wurde, so wird die Transparenz ignoriert und die Hintergrundfarbe laut STYLE verwendet
APPLICATION	Umgebung des Frames mit Vertrauen-Status im Rahmen des Browser-Sicherheitsmodells
ATOMICSELECTION	Attribut wird nicht vererbt an Kinder-Frame
.borderColor	Selektierbarkeit des Objektes einstellen Borderfarbe (Rahmenfarbe) wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no" Eigenschaft .border mit Wert 0
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.contentWindow	Referenz auf das zugehörige Fenster-Objekt laut Collection document.all
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.frameBorder	Borderanzeige ein/aus beim Frame
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.longDesc	Uniform Resource Identifier (URI) zur einer "long description" umwandeln
.marginHeight	Abstand in Pixel vom unteren zum oberen Rand des Objektes
.marginWidth	Abstand in Pixel vom linken zum rechten Rand des Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)



	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.noResize	Frame-Größenänderung ein/aus Größenänderung z.B. durch Maus etc
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrolling	Scrollenbar erzeugen
SECURITY	temporäre Sicherheit des IFRAME wird an Kinder weitervererbt ab IE 6.0
.self	Referenz auf das aktuelle Fenster oder den aktuellen Frame
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein



.clearAttributes()	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.hasChildNodes()	DOM nicht geändert prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert



<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endtag geparkt DOM wird geändert

4.3.2.2.4.3.17. document.frames Collection des Internet Explorer

Feld der Zeiger aller Frames (beim IE inklusive der inline-Frames, also IFRAMES)

Elementefolge laut HTML-Coding

Element ist eine Referenz auf das Fenster mit Frame-Eigenschaften

Syntax:

Achtung: Für eine Referenz auf das Frame-Objekt ohne diese Collection ist immer `document.all` zu kodieren.
Bei Fenstererzeugung bitte jedem Fenster seinen eigenen Namen zuweisen (Name nicht doppelt verwenden)

```
[ var ZeigerAufFeld = ] document.frames
[ var ZeigerAufFeldElement = ] document.frames[Index [ ,SubIndex ] ]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

Beispiel für Inhalte zweier Frames tauschen:

Kodierung im Dokument, dass die beiden Frames definiert !

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
                          logischer_name_rahmen2,html_datei2
                          )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch('r1', 'a.html', 'r2',b.html)'">tauschen</A>
```

Beispiel für Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framennamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument.
Argumententrenner sind Doppelpunkte.

Tausch:

erster Frame retten und dann mit zweiten Frame überschreiben
zweiten Frame mit drittem Frame überschreiben
.....
vorletzten Frame mit letztem Frame überschreiben
letzten Frame mit geretteten ersten Frame überschreiben

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)           // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)      // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);
```



```

for(var i = 0; i < arguments.length; i++)    // Index ab 0
{
    pos_doppelpunkt = arguments[i].indexOf(".");

    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
    {
        // ersten zu ersetzenden Framenamen retten

        if (i=0)
        { var rette_logischer_framename =
            arguments[i].substring(0, pos_doppelpunkt)
          }

        // tauschen der logischen Framenamen
        logischer_framename_zu_ersetzer_framename =
            arguments[i].substring(0, pos_doppelpunkt);

        logischer_framename_ersetzer_framename =
            arguments[i].substring(0, pos_doppelpunkt + 1);

        frames[logischer_framename_zu_ersetzer_framename].location.href =
            logischer_framename_ersetzer_framename;
    }
}

frames[logischer_framename_ersetzer_framename].location.href=
    rette_logischer_framename;
}

}

// -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>

```

Eigenschaften:

.length

Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem()

Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.3.18. frameset Objekt

Hinweis: Wenn ein Fenster mit dem Unterstrichsymbol rechts oben in der Fensterecke minimiert wurde, dann ist es durch Windows in die Hintergrund-Prioritätenfolge eingeordnet worden und arbeitet im Hintergrund bzw. garnicht. Mit anschliessendem Maximieren (Symbol in der Fensterleiste rechts oben) wird das Fenster wieder angezeigt und das geladene Dokument **erneut** geöffnet. Konsequenz daraus ist, dass dieses Maximieren einem Neustart des Dokumentes entspricht, auch wenn der Programmierer diesen Zustand nicht erwünscht. Sound, der z.B. mit Start des Dokumentes erzeugt wird, erklingt erneut mit Maximierung nach einer Fensterminimierung. Analogon ist die Fenstereinstellung im Browser durch z.B. Ein- und Ausblenden der Favoritenleiste. Dieses Verhalten des Fensters mit seinem Dokument ist aber **nicht identisch** mit dem Verhalten nach einer Fenstergrößenänderung z.B. durch Rahmenverschiebung (resize). Die Aktionen des Fensters und seines Dokumentes bezüglich Resize müssen programmiert werden, sind also nicht standardmäßig vorhanden - im Gegensatz zum Verhalten mit/nach Fensterminimierung/-maximierung per Symbole in der rechten oberen Ecke der Fensterleiste. Sollte ein Frameset minimiert werden, dann wird das für den gesamten Frameset getan (Frameset hat 1 Fenster für alle Frames gesamt), allerdings mit Maximierung wird auch das Frameset-Dokument neu geladen. Wichtig dabei sind für Zeiger-Bezüge der Frames auf den Frameset per parent-Zeiger, dass die Daten im Frameset-Dokument mit Maximierung initialisiert werden und somit ebenfalls Daten der Frames, die im Frameset-Dokument abgelegt wurden, also z.B. Daten, die Verhaltensweisen der Frames vor einer Änderung der Frames-Inhalte gespeichert haben.

Erzeugung unter HTML:

```

<FRAMESET
    ROWS="zeilen_liste" oder COLS="spalten_liste"    // Zeilenliste=Liste der Framehöhen
                                                    // mit Kommatrennung
                                                    // Spaltenliste=Liste der Framebreiten
                                                    // mit Kommatrennung
    onLoad="eventhandler1"                        // Anweisungen aktiviert NACH laden ALLER Frames
    onUnload="eventhandler2"
    onerror="eventhandler3"
    onBlur="eventhandler4"
    onFocus="eventhandler5"
>

[<FRAME
    SRC="frame_url"    // url der HTML-Seite als Frameinhalt
    NAME="logischer_frame_name"

```



```

    >]
    ....
    </FRAME>
    .....
</FRAMESET>

```

Rahmen zwischen den einzelnen Fenstern des Framesets entfernen:

```

<FRAMESET      BORDER="0"
                FRAMEBORDER="0"
                FRAMESPACING="0"
                .....
>

```

Hinweise:

BORDER	bei Netscape Breite des Rahmens >= 0 Pixel	
FRAMEBORDER	bei IE Rahmenanzeige 0 oder "no" 1 oder "yes"	aus ein
FRAMESPACING	bei IE Rahmenbreite >=0 Pixel	

Zugriff:

```

document.ID_Frameset.eigenschaft
document.ID_Frameset.methode()

```

Möglichkeiten des Laden eines Dokumentes:

Laden eines fremden Dokumentes innerhalb eines Frame ist rechtlich nicht zulässig (Abmahnungsgefahr), wenn der Eigentümer der Fremddate nicht explizit zugestimmt hat.

Das Laden einer neuen Webseite bei vorhandenem Frameset führt nur dann zur Darstellung außerhalb des Framesets, wenn ein Link z.B. <A> benutzt und dort ein **neues** Fenster geöffnet wird (TARGET-Attribut). Mit anderen Worten: Ein instanzierter Frameset ist z.B. per window.location.href oder window.location.replace() **nicht** aufhebbar, auch wenn das neue Dokument selbst einen Frameset instanziiert. Im letzteren Fall passiert optisch gesehen die Überlagerung des alten mit dem neuen Frameset. Allerdings hat der Programmierer mit Javascript ein riesen Problem: Der Zeiger parent als Zeiger auf dasjenige Dokument, das den Frameset deklariert, zeigt **nicht** auf das neu geladene Dokument mit neuem Frameset ! Daher werden alle Versuche, per Zeiger parent sich auf den neuen Eltern-Frameset zu beziehen, stets beim **vorherigen** Frameset landen und **nicht** beim neuen Dokument, das damit seine eigenen Frames nicht kennt. Somit werden nur Instanzen des alten Framesets vererbt, der auch die Kinder des neuen Frameset kennt, jedoch diese **ihn nicht**. Denn woher sollten die Kinder des neuen Framesets wissen, dass vorher bereits ein Frameset existiert und selbst wenn, sie können mit den Zwangs-Eltern in Form des alten Frameset nichts anfangen. Das Setzen des Zeigers parent auf den null-Wert wird vom Browser bemängelt. Die Verwendung des ID-Attributes im FRAMESET und das Null-Setzen per ID wird vom Browser akzeptiert, aber nicht ausgeführt. Das Schliessen des Eltern-Fensters per self.close() vor dem Umleiten per window.location.href etc. wird ignoriert. Der Zeiger parent ist also nicht umzubiegen.

Einen angenehmen Effekt für den Programmierer hat der Frameset: Wer versucht, sich den Quelltext per Browsermenü Ansicht sich näher zu bringen, wird nur den Quellcode des FRAMESET-Dokumentes zu sehen bekommen.

Laden eines fremden Dokumentes ohne Framedarstellung:

```

<HTML>
  <HEAD>
    <SCRIPT LANGUAGE="JavaScript">
    <!--
      function laden()
      { location.href = "http://www.test.de"; }
    // -->
    </SCRIPT>
  </HEAD>
  <BODY>
    <FORM>
      <INPUT TYPE="button"
        VALUE="www.test.de anwählen "
        onclick="laden()">
    </FORM>
  </BODY>
</HTML>

```

Eigenes Dokument wird durch fremde Webseite geladen:



CopyRight-Meldung auf fremden Host erzeugen:

Beispiel 1:

```
// In diesem Beispiel schreibt das Script Text auf die "entführte" Seite!
var HostUrl="www.test.de";

if (    (parent !=null)
    &&  (parent != self)
    )
{
    var host=parent.location.hostname;

    if(host != HostUrl)
    {
        document.write(
            "Diese Seite wurde ausgeliehen bei "
            + "<A HREF=\"" + location.href
            + "\" TARGET='_parent'"
            + ">"
            + HostUrl
            + "</A>"
        );
    }
}
```

Beispiel 2:

```
<BODY>
.....
if (    (parent != null)
    &&  (parent != self)
    )
{
    var  meine_url = "www.test.de"
    var  mein_host_name = "http://" + meine_url;

    var  fremder_host_name = parent.location.hostname;

    if ( fremder_host_name != mein_host_name)
    {
        document.write(      " Diese Seite liegt auf "
                               + "<A HREF=\"" + location.href + "\"\"
                               + " TARGET=\"_parent\""
                               + ">"
                               + "</A>"
                               + " und stammt von "
                               + mein_host_name
        );
    }
}
.....
</BODY>
```

Framedarstellung der eigenen Webseite sofort und ohne Bildschirmmeldung aktivieren:

Dieses Coding muss in jedes HTML-Dokument, das in einem Frame geladen wird. Bei Einzelaufwurf des Dokumentes wird automatisch die Seite mit dem FRAMSET aktiviert, also die vollständige Framedarstellung.

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
    var EigenerHost="http://www.test.de";
        // window.location.hostname ist Leerkette, wenn Browser offline
    var StartSeite="_start.html";

    function InaktiveFrameDarstellungAktivieren()
    {
        if (top.frames.length == 0)
        {
            // keine Frame-Darstellung aktiv, also diese aktivieren
```



```

        top.location.href = Startseite; // muss das FRAMESET enthalten !
    }
}

if ( (parent != null) // Elternobjekt existiert
    && (parent != self) // Eltern sind vorhanden Eltern: dieses Dokument (self) ist ein Kind
    )
{
    // aktuellen ElternHost ermitteln
    var ElternHost=parent.location.hostname;

    // ElternHost prüfen ob eigener und nicht leer, also Browser online ist
    if ( (ElternHost != "") // Browser ist online
        && (ElternHost != EigenerHost)
        )
    {
        // fremder Host, also als oberstes Fenster nun die Startseite anzeigen
        // und damit den eigenen Host aktivieren
        top.location.href=EigenerHost + '/' + Startseite;
    }
    else
    {
        // eigener Host und/oder Browser ist offline
        InaktiveFrameDarstellungAktivieren();
    }
}
else
{
    // Elternobjekt existiert nicht und/oder dieses Dokument (self) ist kein Kind
    InaktiveFrameDarstellungAktivieren();
}

//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Reload eines Dokumentes mit allen seinen FRAMES:

```

function frame_neu_laden()
{
    for (var i=0; i<windows.FRAMES.length; i++)
    { windows.frames[i].location.reload(false); }
}

<FRAMESET onResize="frame_neu_laden()">

```

Datei ohne FRAMESET in eine Datei mit FRAMESET laden

```

if (self.location == top.location)
{ location.href="/FRAMESET.html?" + escape(location.pathname); }

```

Mehrere Frameinhalte gleichzeitig ändern:**Inhalte zweier Frames tauschen:**

Kodierung im Dokument, dass die beiden Frames definiert !

```

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch(logischer_name_rahmen1,html_datei1,
        logischer_name_rahmen2,html_datei2
        )
    {
        frames[logischer_name_rahmen1].location.href = html_datei1;
        frames[logischer_name_rahmen2].location.href = html_datei2;
    }
// -->
</SCRIPT>

<A HREF="javascript:parent.rahmentausch(0, 'a.html', 1,b.html)">tauschen</A>

```

Inhalte beliebig vieler Frames als Ring tauschen:

Die logischen Framenamen werden als variable Argumenteliste übergeben und dienen der Indizierung der Frame im Dokument. Argumententrenner sind Doppelpunkte.

Tausch:

```

erster Frame retten und dann mit zweiten Frame überschreiben
zweiten Frame mit drittem Frame überschreiben
.....
vorletzten Frame mit letztem Frame überschreiben
letzten Frame mit geretteten ersten Frame überschreiben

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    function rahmentausch()
    {
        if (arguments.length>1)          // Länge ab 1, mindestens 2 Argumente
        {
            var pos_doppelpunkt = arguments[0].indexOf(":");

            if (pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
            {
                var rette_logischer_framename= arguments[0].substring(0, pos_doppelpunkt);

                for(var i = 0; i < arguments.length; i++)    // Index ab 0
                {
                    pos_doppelpunkt = arguments[i].indexOf(":");

                    if(pos_doppelpunkt != -1)    // nächstes Argument ist vorhanden
                    {
                        // ersten zu ersetzenden Framenamen retten

                        if (i=0)
                        { var rette_logischer_framename =
                            arguments[i].substring(0, pos_doppelpunkt)
                        }

                        // tauschen der logischen Framenamen
                        logischer_framename_zu_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt);

                        logischer_framename_ersetzender_frame =
                            arguments[i].substring(0, pos_doppelpunkt + 1);

                        frames[logischer_framename_zu_ersetzender_frame].location.href =
                            logischer_framename_ersetzender_frame;
                    }
                }

                frames[logischer_framename_ersetzender_frame].location.href=
                    rette_logischer_framename;
            }
        }
    }
// -->
</SCRIPT>
<A HREF="javascript:parent.rahmentausch('r1:a.html', 'r2:b.html', 'r3:c.html')">tauschen</A>

```

Frame und Datenaustausch:

Zwischen Frames können Daten ausgetauscht werden.

Beispiel Adressierung eines Frame über das FRAMESET

```

<FRAMESET .... >
    <FRAMESET ..... >
        <FRAME SRC="test.html" NAME="test">
        <FRAME SRC="test1.html" NAME="test1">
    </FRAMESET>
    <FRAME SRC="test2.html" NAME="test2">
</FRAMESET>

parent.test2.location.href="neu.html";    // Laden von neu.html in den Frame test2

```

Beispiel: Auf Objekte im FRAMESET-Dokument durch ein FRAME-Dokument zugreifen



im FRAMESET-Dokument wird kodiert `var Kette="Hallo !";`

im FRAME-Dokument wird auf Kette zugegriffen: `alert(parent.Kette);`

Beispiel: Auf Objekte im FRAME-Dokument durch das FRAMESET-Dokument zugreifen

im FRAME-Dokument wird kodiert `var Kette="Hallo !";`
 `<FRAME ...NAME ="test2" ...>`

im FRAMESET-Dokument wird auf Kette zugegriffen: `alert(parent.test2.Kette);`

Beispiel: Textdaten-Übergabe durch an Url angehängte Textdaten

quelle.htm: **lädt ziel.htm**
 übergibt Textdaten an ziel.htm

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function ziel_seite_laden_mit_datenuebergabe(uebergabe_daten)
    {location.href = "ziel.htm?" + escape(uebergabe_daten);}
      // Url von ziel.htm als Suchbegriff mit angehangenen Daten merken, die per escape() in das
      //      Url-Format konvertiert wurden
// -->
</SCRIPT>
</HEAD>

<BODY>
An ziel.htm zu &uuml;bergabenden Text eingeben:
<FORM>
    <TEXTAREA      NAME=eingabe
                      ROWS=5
                      COLS=40
    >
    </TEXTAREA>
    <BR>
    <INPUT  TYPE=button
            VALUE="Zielseite laden"
            onClick=" ziel_seite_laden_mit_datenuebergabe(this.form.eingabe.value)">

</FORM>
</BODY>
</HTML>
```

ziel.htm: **wird durch quelle.htm geladen**
 erhält Textdaten von quelle.htm

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    datenuebernahme()
    {
        // Url mit angehangenen Daten holen
        // search liefert ?daten
        uebernahme_daten= location.search;

        // ? abschneiden
        uebernahme_daten= uebernahme_daten.substring(1, uebernahme_daten.length);

        // unescape: von Url-Format nach Zeichenkette
        document.ausgabe.ausgabefeld.value=unescape(uebernahme_daten);
    }

// -->
</SCRIPT>
</HEAD>

<BODY onLoad=" datenuebernahme ()">
Von quelle.htm &uuml;bernommener Text:
    <FORM NAME=ausgabe>
        <TEXTAREA      NAME=ausgabefeld
                        ROWS=10 COLS=40>
```



```

        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Beispiel: Textdaten-Übergabe durch Fenster-Handle

Die Textdaten und die der Javascript-Code, der die Textdaten verarbeitet, sind **getrennte** HTML-Dokumente. Nachteil dieser Variante ist, dass in beiden Dokumenten die logischen Namen vom Formular und dem Textarea identisch

kodiert werden müssen, also eine doppelte Verwaltung nötig ist.

HTML-Dokument, das die Textdaten enthält:

```

<HTML>
<BODY>
    <FORM NAME=formular>
        <TEXTAREA NAME=text_area>
            // Hier die Textdaten als Wert von TEXTAREA
        </TEXTAREA>
    </FORM>
</BODY>
</HTML>

```

Dokument, das die Textdaten verarbeitet:

```

<HTML>
<HEAD>
    <SCRIPT LANGUAGE="JavaScript">
<!--
    function TextAreaWertLesen(url)
    {
        // url enthält den Pfad und den Namen des Dokumentes, dass die
        // Textarea-Werte enthält
        var handle = window.open(url_der_datendatei,"", "width=100,height=100");

        // Handle erzeugen und Fenster mit dem Textarea anzeigen
        // (Fenstergröße ist egal)
        var data = handle.document.forms[formular].elements[text_area].value;

        handle.close();

        return data;
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

frameset Objekt des Internet Explorer:

Container aller Frames

Die Collection frames wird unter document.frames beschrieben !

Das Laden einer neuen Webseite bei vorhandenem Frameset führt nur dann zur Darstellung außerhalb des Framesets, wenn ein Link z.B. <A> benutzt und dort ein **neues** Fenster geöffnet wird (TARGET-Attribut). Mit anderen Worten: Ein instanzierter Frameset ist z.B. per window.location.href oder window.location.replace() **nicht** aufhebbar, auch wenn das neue Dokument selbst einen Frameset instanziiert. Im letzteren Fall passiert optisch gesehen die Überlagerung des alten mit dem neuen Frameset. Allerdings hat der Programmierer mit Javascript ein riesen Problem: Der Zeiger parent als Zeiger auf dasjenige Dokument, das den Frameset deklariert, zeigt **nicht** auf das neu geladene Dokument mit neuem Frameset ! Daher werden alle Versuche, per Zeiger parent sich auf den neuen Eltern-Frameset zu beziehen, stets beim **vorherigen** Frameset landen und **nicht** beim neuen Dokument, das damit seine eigenen Frames nicht kennt. Somit werden nur Instanzen des alten Framesets vererbt, der auch die Kinder des neuen Frameset kennt, jedoch diese **ihn nicht**. Denn woher sollten die Kinder des neuen Framesets wissen, dass vorher bereits ein Frameset existiert und selbst wenn, sie können mit den Zwangs-Eltern in Form des alten Frameset nichts anfangen. Das Setzen des Zeigers parent auf den null-Wert wird vom Browser bemängelt. Die Verwendung des ID-Attributes im FRAMESET und das Null-Setzen per ID wird vom Browser akzeptiert, aber nicht ausgeführt. Das Schliessen des Eltern-Fensters per self.close() vor dem Umleiten per window.location.href etc. wird ignoriert. Der Zeiger parent ist also nicht umzubiegen. Die Anwendung eines Zeigerbezuges vom alten Frameset über den neuen Frameset auf die Kinder des neuen Frameset, also die Verkettung von parent-Zeigern, ist ziemlich unsinnig, da parent ansich der Zeiger auf das aktuelle Frameset sein sollte, auch wenn in der Objekthierarchie der neue Frameset ein Kind vom alten Frameset sein müsste. Alternativ zu dieser unangenehmen Framesetüberlagerung sind Konstruktionen von DIV-Objekten innerhalb eines **gemeinsamen** Dokumentes möglich, wobei in die DIV's dann je ein Dokument geladen wird (document.open()). Es können dann alle Dokumente direkt über das ID des jeweiligen DIV's adressiert werden, allerdings haben diese DIV's eben **keine** Fenstereigenschaften wie Frames im gemeinsamen Frameset-Fenster. (DIV- und FRAME/IFRAME-Objekte unterscheiden sich in Eigenschaften und Methoden). So gesehen stellt document.open() eine eingeschränkte Lösung dar. HTML-orientiert ist die Nutzung eines



Links, z.B. <A> mit Targetwert _top, also dem Zeiger auf das oberste Fenster, in dem der alte Frameset sitzt. Nur ist ein Link in der Regel visuell vorhanden und muss durch den User aktiviert werden. Letzte Alternative ist die Umlenkung per META-Tag anhand eines Timers.

Hinweis: Wenn ein Fenster mit dem Unterstrichsymbol rechts oben in der Fensterecke minimiert wurde, dann ist es durch Windows in die Hintergrund-Prioritätenfolge eingeordnet worden und arbeitet im Hintergrund bzw. garnicht. Mit anschließendem Maximieren (Symbol in der Fensterleiste rechts oben) wird das Fenster wieder angezeigt und das geladene Dokument **erneut** geöffnet. Konsequenz daraus ist, dass dieses Maximieren einem Neustart des Dokumentes entspricht, auch wenn der Programmierer diesen Zustand nicht erwünscht. Sound, der z.B. mit Start des Dokumentes erzeugt wird, erklingt erneut mit Maximierung nach einer Fensterminimierung. Analogon ist die Fenstereinstellung im Browser durch z.B. Ein- und Ausblenden der Favoritenleiste. Dieses Verhalten des Fensters mit seinem Dokument ist aber **nicht identisch** mit dem Verhalten nach einer Fenstergrößenänderung z.B. durch Rahmenverschiebung (resize). Die Aktionen des Fensters und seines Dokumentes bezüglich Resize müssen programmiert werden, sind also nicht standardmäßig vorhanden - im Gegensatz zum Verhalten mit/nach Fensterminimierung/-maximierung per Symbole in der rechten oberen Ecke der Fensterleiste. Sollte ein Frameset minimiert werden, dann wird das für den gesamten Frameset getan (Frameset hat 1 Fenster für alle Frames gesamt), allerdings mit Maximierung wird auch das Frameset-Dokument neu geladen. Wichtig dabei sind für Zeiger-Bezüge der Frames auf den Frameset per parent-Zeiger, dass die Daten im Frameset-Dokument mit Maximierung initialisiert werden und somit ebenfalls Daten der Frames, die im Frameset-Dokument abgelegt wurden, also z.B. Daten, die Verhaltensweisen der Frames vor einer Änderung der Frames-Inhalte gespeichert haben.

Zugriff auf Frameset-Eigenschaft:

Beispiel

```
var Wert = document.all.ZeigerAufFrameset.eigenschaft;
```

mit ZeigerAufFrameset laut ID-Attribut

```
<FRAMESET ID="ZeigerAufFrameset" ..... >
```

Eigenschaften:

ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.border	Rahmendicke in Pixel
.borderColor	Borderfarbe (Rahmenfarbe)
	wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no"
	Eigenschaft .border mit Wert 0
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.cols	Breite aller Frames eines Frameset (Breite des Frameset als Container)
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.frameBorder	Borderanzeige ein/aus beim Frame
.frameSpacing	Zwischenraum in Pixel zwischen 2 benachbarten Frames
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
	Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!)
	muss beim Formular für alle zu sendenden Felder kodiert sein !!
	Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag
	wirksam mit parsen des Ende-Tags
	nur nach kompletten Einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren



.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.rows	Höhe aller Frames eines Frameset (Höhe des Frameset als Container)
.scopeName	Namensraum laut XMLNS-Attribut
.sourceIndex	Index des Objektes in der Collection document.all
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes
	Anspringen verbunden mit Focus erhalten
	--> Ereignisse werden ausgelöst !!
	unter IE 5.x
	onblur, onfocus
	ab IE 5.x
	onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste
	für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste
	für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
	DOM wird geändert
	Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
	Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
	vor IE 5.0 TABINDEX-Attribut muss kodiert sein
	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen
	außer ID, STYLE und per Script definierte Attribute
	Script-erzeugte Attribute nicht entfernen
.cloneNode()	DOM wird geändert
	Objekt klonen und Referenz des erzeugten Klone liefern
	DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
	auch für CSS-Layout
	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint()
	also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben
	Overbereich der Maus ist mehr als 1 Pixel gross
	beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
	DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler
	wobei Registrierung mit Methode .attachEvent() aktiviert wurde
	Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt



	(also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByTagName()
.hasChildNodes()	DOM nicht geändert prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern



.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.setActive()	DOM wird geändert Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.swapNode()	DOM wird geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt

4.3.2.2.4.3.19. document.html Objekt des Internet Explorer

Das Objekt repräsentiert den HTML-Tag.

Scriptsteuerung erst ab IE 4.x

sonst siehe HTML-Dokument mit Scriptcode

Beispiel

```
<HTML>
<BODY>
  <P>This is an HTML document.</P>
</BODY>
</HTML>
```

Eigenschaften:

.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)



	nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scroll	Scrollbar-Anzeige ein/aus vor IE 6.x nur BODY-Objekt ab IE 6.x alle HTML-Elemente wenn DOCTYPE im Kopf des Dokumentes kodiert
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.version	Document Type Definition-Version (DTD-Version) ab IE 5.x
XMLNS	Namensraum für Benutzer-Tags zu einem HTML-Tag ab IE 5.x nur für das Tag HTML möglich Namensraum: Präfix des Benutzer-Tags oder Uniform Resource Name (URN)

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM- Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos



	erreicht haben
	Überbereich der Maus ist mehr als 1 Pixel gross
	beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
	DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
	DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern
	DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
	DOM nicht geändert
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
	DOM nicht geändert
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
	DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
	DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
	DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
	DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
	DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
	DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
	DOM wird geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt



	ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
	sichtbar erst mit parsen des Endetags
	DOM wird geändert
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen
	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
	nur sichtbar wenn Endetag geparkt
	DOM wird geändert

4.3.2.2.4.3.20. **html comment Objekt des Internet Explorer**

repräsentiert den HTML-Kommentar, der weder geparkt noch gerendert wird

Beispiel

```
<!-- This text will not appear in the browser window. -->
```

Event: nur `onlayoutcomplete`

Eigenschaften:

<code>.innerHTML</code>	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein <code>innerHTML</code> haben, z.B. <code></code>
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
<code>.outerHTML</code>	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag
	wirksam mit parsen des Ende-Tag
	nur nach kompletten Einlesen des Dokumentes nutzbar
<code>.tagName</code>	Tag-Bezeichner des Objektes
<code>.text</code>	Text des HTML-Kommentares

4.3.2.2.4.3.21. **iframe Objekt des Internet Explorer**

ist dem DIV-Objekt sehr ähnlich

ist eine effektivere Variante des FRAME-Objektes

ACHTUNG: Die Ausführung von Scripten etc. im IFRAME können ab IE 6.0 per Browser-Menü "Extra-Internet Optionen-Sicherheit" unter "Programme und Dateien in einem IFRAME starten" durch den User auf deaktivieren gesetzt werden.

Damit ist der IFRAME nicht mehr nutzbar bezüglich Scripts, Protokolle und URL-Aufrufen

z.B. dann im IFRAME nicht ausführbar Code

```
<IFRAME src="http://www.test.de"></IFRAME>
```

Alternativ wird ein neues Fenster geöffnet, wenn

ab IE 4.x

ist ein Dokument in einem Dokument

Eltern des obersten IFRAME ist das BODY-Objekt

benutzt folgende Collectionen `frames`

`document.all`

Die Collection `frames` wird unter `document.frames` beschrieben !

Beispiele

```
<IFRAME ID="ID_Frame" IFrame1 FRAMEBORDER=0 SCROLLING=NO SRC="sample.htm">
  <A HREF="http://www.test.de" TARGET="ID_Frame"> www.test.de </A>
</IFRAME>
```

```
var ZeigerAufDocumentAllImFrame = document.frames("IFrame1").document.all
```

```
var HintergrundFarbeImFrame = document.frames("sFrameName").document.body.style.backgroundColor;
```

```
var BoderWert = document.all.zeiger_auf_frame.style.border;
```

```
var MarginWidthWert = document.all.id_des_frame.marginWidth
```

Erweiterungen ab IE 5.5

Transparenz: möglich, wenn

im IFRAME das Attribut `ALLOWTRANSPARENCY` auf "true" gesetzt ist

```
Bsp.: <IFRAME NAME="Frame1" SRC="frame.htm" ALLOWTRANSPARENCY="true">
      </IFRAME>
```

UND Eltern des IFRAME, z.B. BODY, mit `background-color` auf "transparent" gesetzt ist

```
<BODY STYLE="background-color:transparent">
```

Hinweis: Attribut `BGCOLOR` ist deprecated !!!

IFRAME ohne Fensterrahmen-Elemente möglich, auch mit Überlappung

sind schneller in der Darstellung als FRAME mit Fensterrahmen-Elementen



nutzen weniger RAM
scrollen schneller

Attribut z-index nutzbar für Überlappung von IFRAME

z.B. <IFRAME SRC="frame.htm" STYLE="z-index:1" ></IFRAME>

je höher der z-index, um so höher der IFRAME in der Hierarchie
oberster IFRAME hat höchsten z-index
z-index kann auch < 0 sein

Eigenschaften:

.align	Ausrichtung
.allowTransparency	Transparenz ein/aus
	Transparenz: z.B. Hintergrundfarbe des Elternobjektes wird zur Hintergrundfarbe des Frames
	Achtung: Wenn Frame-eigenes STYLE-Attribut mit Wert für background-color
	also STYLE="background-color: farb_bezeichner"
	kodiert wurde, soe wird die Transparenz ignoriert und die
	Hintergrundfarbe laut STYLE verwendet
APPLICATION	Umgebung des Frames mit Vertrauen-Status im Rahmen des Browser-Sicherheitsmodells
	Attribut wird nicht vererbt an Kinder-Frame
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
	siehe Objekt currTimeState und Behavior .style.time2
.border	Rahmendicke in Pixel
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.contentWindow	Referenz auf das zugehörige Fenster-Objekt laut Collection document.all
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.disabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
	ab IE 6.x
	alternativ: Eigenschaft .dur
	siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.frameBorder	Borderanzeige ein/aus beim Frame
.hasMedia	Objekt ist HTML-Media-Objekt
	siehe Objekt currTimeState und Behavior .style.time2
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt
	und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und
	betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
	Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.longDesc	Uniform Resource Identifier (URI) zur einer "long description" umwandeln
.marginHeight	Abstand in Pixel vom unteren zum oberen Rand des Objektes
.marginWidth	Abstand in Pixel vom linken zum rechten Rand des Objektes
.name	Name des Objektes (nicht ID !!!)
	muss beim Formular für alle zu sendenden Felder kodiert sein !!
	Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)



.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrolling	Scrollenbar erzeugen
SECURITY	temporäre Sicherheit des IFRAME wird an Kinder weitervererbt ab IE 6.0
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currentTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currentTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern



	DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar
.cloneNode()	DOM wird geändert Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.detachEvent()	DOM nicht geändert Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getElementsByTagName()	DOM nicht geändert Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.hasChildNodes()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten



	(Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient



Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 .swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.22. img Objekt

Erzeugung unter HTML:

Beispiele:

Bild nicht im Formular:

```
<IMG SRC="url_des_bildes_in_voller_auflösung"
NAME="logischer_bild_name"
LOWSRC="url_des_bildes_in_geringer_auflösung"
ALT="alternativer_text"
ALIGN=ausrichtung
HEIGHT=hoehe
WIDTH=breite
BORDER=rahmenbreite
HSPACE=abstand_links_und_rechts_zur_umgebung
VSPACE=abstand_oben_und_unten_zur_umgebung
ISMAP
USEMAP= "map_url#image_map"
oder "map_name"
onAbort="eventhandler1"
onerror="eventhandler2"
onLoad="eventhandler3"
>
```

Bild im Formular:

```
<INPUT TYPE=image
SCR=url_des_bildes_in_voller_auflösung"...
>
```

Beispiel für Linie mit variabler Dimension:

Es wird ein Bild aus 1x1 neu dimensioniert, das eine kleine Dateigröße hat und nur 1 mal geladen werden muss.
 Durch die Angaben von Breite und Höhe kann eine vertikale oder horizontale Linie erzeugt werden. Die
 Linienpositionierung erfolgt per STYLE-Attribut.

horizontale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="300" HEIGHT="10" BORDER="0" ALT="" STYLE=" ....">
```

vertikale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="10" HEIGHT="300" BORDER="0" ALT="" STYLE=" ....">
```

Erzeugung in Script:

```
var Bild = new Image([breite, hoehe]);
```

erzeugt Instanz und lädt zugleich das Bild
 zeigt das Bild NICHT an

Verwendung z.B. bei animiertem Bild, wobei die Animation geladene Bilder voraussetzt

Zugriff:

Bild nicht im Formular:

```
document.ID_Img.eigenschaft
document.images[index].eigenschaft
```

index: ab 0
 muss in [] kodiert sein

Bild im Formular:

```
document.ID_Img.eigenschaft
document.images[index].eigenschaft
document.ID_Formular.elements[index].eigenschaft
```

index: ab 0
 muss in [] kodiert sein

Bild in Script: per Variable aus new

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```
var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url; // ohne "" kodieren da Zeiger
document.write(' <IMG NAME="IMG_Bild"
```




```

+ ' SRC=' + BildObjekt.src + ''
+ ' HEIGHT=' + Bild_Hoehe
+ ' WIDTH=' + Bild_Breite
+ '>'
);

```

Eigenschaften (ausgewählte):

.border	entspricht BORDER nur lesen
.complete	wenn mit true belegt, so Bild komplett geladen wenn mit false belegt, so Bild noch nicht komplett geladen nur lesen
.height	entspricht HEIGHT nur lesen
.hspace	entspricht HSPACE nur lesen
.lowsrc	entspricht LOWSRC
.name	entspricht NAME nur lesen
.src	entspricht SRC
.vspace	entspricht VSPACE; Standard ist 0 nur lesen
.width	entspricht WIDTH nur lesen

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```

var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url; // ohne "" kodieren da Zeiger
document.write(
+ '<IMG NAME="IMG_Bild"'
+ ' SRC=' + BildObjekt.src + ''
+ ' HEIGHT=' + Bild_Hoehe
+ ' WIDTH=' + Bild_Breite
+ '>'
);

```

Events bei sich überlagernden Objekten:

Beispiel: Sollte ein Image mit seiner Erzeugung ein anderes Image überlagern, so ist die Eventübergabe von und an das untere Image unterbrochen: Ereignisse onXXX kommen nicht mehr durch, solange das untere Bild nicht den **Fokus** erhält. Alternativ ist die Style-Eigenschaft z-index zu kodieren und dann der z-index auf den obersten zu setzen.

img Objekt des Internet Explorer:

HTML-Container für ein Bild, Videoclip, VRML-Datei
Scriptsteuerung erst ab IE 4.x
Anwendung auch im Formular per input image Objekt
Die Collection images wird unter document.images beschrieben !

ab IE 5.x werden folgende Image-Dateiarten unterstützt:

*.avi	Audio-Visual Interleaved (AVI)
*.bmp	Windows Bitmap (BMP)
*.emf	Windows Enhanced Metafile (EMF)
*.gif	Graphics Interchange Format (GIF)
*.jpg, *.jpeg	Joint Photographic Experts Group (JPEG)
*.mov	Apple QuickTime Movie (MOV)
*.mpg, *.mpeg	Motion Picture Experts Group (MPEG)
*.png	Portable Network Graphics (PNG)
*.wmf	Windows Metafile (WMF)
*.xbm	X Bitmap (XBM)

Hinweis: Ereignis onfocus nicht ausgelöst bei einem MAP-Image
Kodierung der Datei-Url:
für statisches Bild das Attribut SRC verwenden
für Videoclip oder VRML-Datei (virtual reality modeling language-Datei) das Attribut DYN SRC verwenden

Beispiel

Beispiel für Aus- und Einblende eines Bildes:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
var Sichtbar=true; // DIV ist natürlich nach dem Dokument-Laden sichtbar
var StandardFarbe="green";

```



```

function Blenden()
{
    // Fade zurücksetzen
    DIV_ID.filters[0].Apply();

    if (Sichtbar)
    {
        Sichtbar=false;
        DIV_ID.style.visibility="hidden";
    }
    else
    {
        Sichtbar=true;
        DIV_ID.style.visibility="visible";
        DIV_ID.style.backgroundColor=StandardFarbe;
    }

    // Fade starten
    DIV_ID.filters[0].Play();
}

// -->
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="Blenden()">
    Aus- und Einblenden
</BUTTON>
<BR>
<BR>
<DIV ID="DIV_ID"
    STYLE="height:250px;width:250px;background-color:red;
        filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
    // alles EINE Zeile !!
>
    <IMG SRC="Bild.gif" ....HEIGHT=250 WIDTH=250>
    // Bild-Dimension muss in die DIV-Dimension reinpassen !

</DIV>
</BODY>
</HTML>

```

Beispiel für Aus- und Einblende von Bildern mit Bildwechsel:

Variante 1

Es werden zwei DIV an absoluter Position überlagert. Jedes DIV hat einen eigenen Inhalt, hier im Beispiel sein eigenes Bild. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der beiden Bilder erreicht. Pro Bild wird ein eigenes DIV erzeugt.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var DIV_ID1_Sichtbar=true; // DIV_ID1 ist nach dem Dokument-Laden sichtbar
                                // DIV_ID2 aber nicht

    function BildWechsel()
    {
        // Fade zurücksetzen
        DIV_ID1.filters[0].Apply;
        DIV_ID2.filters[0].Apply;

        if (DIV_ID1_Sichtbar)
        {
            DIV_ID1_Sichtbar = false;
            DIV_ID1.style.visibility="hidden";
            DIV_ID2.style.visibility="visible";
        }
        else
        {
            DIV_ID1_Sichtbar = true;
            DIV_ID1.style.visibility="visible";
            DIV_ID2.style.visibility="hidden";
        }

        // Fade starten
    }

```




```

        DIV_ID1.filters[0].Play();
        DIV_ID2.filters[0].Play();
    }
    // -->
</SCRIPT>
</HEAD>
<BODY>
    <BUTTON onclick="BildWechsel()">
        Bild wechseln mit Aus- und Einblenden
    </BUTTON>
    <BR>
    <BR>
    <DIV ID="DIV_ID1"
        STYLE="height:250px;position:absolute;top:50;width:250px;
            background-color:red; visibility:visible
            filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile
    >
        <IMG SRC="DIV_ID1_Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
    </DIV>

    <DIV ID="DIV_ID2"
        STYLE="height:250px;position:absolute;top:50;width:250px;
            background-color:red; visibility:hidden
            filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile
    >
        <IMG SRC="DIV_ID2_Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
    </DIV>
</BODY>
</HTML>

```

Variante 2

Es wird der innere DIV-Bereich zwischen <DIV> und </DIV> ersetzt und damit je ein Bild dargestellt. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der Bilder erreicht.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var BildUrlFeld=Array
    (
        // hier beliebig viele Bilder eintragen
        "test1.jpg",
        "test2.jpg",
        "test3.jpg"
    );

    var BildUrlFeld_Laenge= BildUrlFeld.length;

    //      fuer alle Bilder die identischen Dimensionen !!!
    var BildHoehe=444;
    var BildBreite=640;

    // Zeigerfeld zum Vorladen der Grafiken
    var BildZeigerFeld = Array();

    var Index=1;          // Starten mit test2.jpg, da test1.jpg bereits angezeigt mit Laden des DIV

    function BildWechsel()
    {
        // Fade zurücksetzen
        DIV_ID.filters[0].Apply();

        // Bild im DIV anzeigen
        document.AktuellesBild.src= BildZeigerFeld[Index].src;

        // Fade starten
        DIV_ID.filters[0].Play();

        // nächstes Bild einstellen
        Index++;
    }

```



```

        // Index korrigieren
        if (Index >= BildUrlFeld_Laenge)
        {Index=0;}
    }

    // nachfolgender Code wird mit dem Laden des HEAD-Teiles abgearbeitet

    // Bildobjekte vorladen: allokalieren und Zeiger merken per Prototyping
    for (i=0; i<= BildUrlFeld_Laenge; i++)
    {
        // Image-Zeiger bilden als Feldeintrag
        BildZeigerFeld[i] = new Image();

        // Url dem Zeiger zuweisen und Bild somit vorladen
        BildZeigerFeld[i].src= BildUrlFeld[i];
    }

    // DIV erzeugen mit test1.gif als Startbild
    //                               in Dimension aller Bilder
    document.write(
        '<DIV '
        +   'ID="DIV_ID" '
        +   'STYLE="height: ' + BildHoehe + 'px;width: ' + BildBreite + 'px;'
        +   'filter:progid:DXImageTransform.Microsoft.Fade(duration=1);'
        +   '""
        +   '>\n'
    );

    document.write(
        '<IMG NAME="AktuellesBild"'
        +   'SRC="" + BildZeigerFeld[0].src + "'
        +   'WIDTH=' + BildBreite + "'
        +   'HEIGHT=' + BildHoehe
        +   '>\n'
    );

    document.write(
        '</DIV>\n');

    // Button zum Bildwechsel anzeigen
    document.write(
        '<BUTTON onclick="BildWechsel()">'
        +   'Naechstes Bild'
        +   '</BUTTON>\n'
    );

    // -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Beispiel für mehrere Bilder in den RAM laden und mit Wartezeit preloaden:

```

<HTML>
<HEAD>
    <SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
    <!--
        var bild_feld = [
            "b1.gif",
            // hier alle Bilder eintragen.
        ];

        var wartezeit = 500; // Zeit in ms zwischen zwei Ladevorgaengen
        var feld_index = 0;

        function preloaden()
        {
            var bild = new Image();
            bild.src = bild_feld[feld_index];

            feld_index ++;

            if(feld_index < bild_feld.length) // Länge ab 1, Index ab 0
            {setTimeout('preloaden()', load_next)}
        }

        function start()

```



```

        {setTimeout('preloaden()', wartezeit)}
    // -->
</SCRIPT>
</HEAD>
<BODY ... onLoad="start()">
</BODY>
</HTML>

```

Beispiele für Bildfolge:

Beispiel 1:

10 Bilder mit Namen b1.gif bis b10.gif. Wichtig ist der numerische Namenteil 1 bis 10, da als Zähler verwendet

```

<HTML>
<HEAD>
    <SCRIPT LANGUAGE="JavaScript">
    <!--

        verzoegerung = 120; // in Millisekunden
        bildNummer = 2;

        bilder= new Array(); //nimmt die Bilder der Animation auf

        // hier werden die Bilder im Hintergrund geladen
        for (i = 1; i <= 10; i++)
        {
            bilder[i] = new Image();
            bilder[i].src = "b" + i + ".gif";
        }

        function naechstesBild()
        {
            document.animation.src = bilder[bildNummer].src;
            bildNummer++;
            if (bildNummer > 10) bildNummer = 1;
        }

    // -->
    </SCRIPT>
</HEAD>

<BODY>
    <IMG SRC="dp1.gif"
        NAME="animation"
        WIDTH="165"
        HEIGHT="185"
        onLoad="setTimeout('naechstesBild()', verzoegerung)">

</BODY>
</HTML>

```

Beispiel 2:

Es können **geladene Bilder** bild1.gif bis bild5.gif an fester Position im Wechsel ausgegeben werden.
 geladenen Bilder erzeugen per Image-Objekt
 erstes Bild wird angezeigt per
 alle Bilder müssen gemeinsame Dimension haben, sonst wird gestreckt oder gestaucht
 Durch Aktualisierung vom Attribut SRC aus mit der jeweiligen Url des Bildes wird der sichtbare Bildwechsel vollzogen.

```

<HEAD>
<SCRIPT ...>
<!--    // alle Bilder vorladen

        // globale Größen festlegen

        var bild_anzahl=5;                // ab 1

        var feld=new Array(bild_anzahl);    // Inhalt des Feldes ist hier noch unbekannt

        feld[1]=new Image();                // Feldelement 1 ist Image-Objekt
        //                                Bild 1 wird vorgeladen und nicht nicht agenzeigt,
        //                                da der Javascript-Code VOR dem IMG-Tag
        //                                aus <BODY> .. </BODY> abgearbeitet wird !

        feld[1].url="bild1.gif";            // Prototyping: Eigenschaft url hinzufügen für Index 1
        ....
    
```



```

feld[5]=new Image();           // Feldelement 5 ist ein Image-Objekt
feld[5].url="bild5.gif";       // Prototyping: Eigenschaft url hinzufügen für Index 5

var index=bild_anzahl;         // auch möglich           var index= feld.length;

function bild_wechsel()
{
    // Nummer des aktuellen Bild ermitteln
    if (index == bild_anzahl)
    {index=0;}

    index+=1;

    // Kopieren der Url nach <IMG>-Attribut SRC
    //      also gleichzeitig anzeigen
    self.document.logischer_img_name.url=feld[index].url;
}
// -->
</SCRIPT>
</HEAD>
<BODY>
    <IMG      NAME="logischer_img_name"
              SRC="bild1.gif"           // vorgeladenes Bild anzeigen
    >
    <SCRIPT>
    <!--
        setInterval(bild_wechsel,2000); // periodische Abarbeitung alle 2 Sekunden
    // -->
    </SCRIPT>
</BODY>

```

Beispiele für festes, nicht mitscrollendes Bild:

Beispiel 1:

festes Grafik in einem Fenster für Internet Explorer oder Netscape

```

<HTML>
<HEAD>
<TITLE> .... </TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
    posX=20;           // horizontal
    posY=20;           // vertikal
    idName="grafik_name";

    function merkeYPosition()
    {
        if (document.all)           // IE
        {merkeY=document.body.scrollTop;}
        else                         // Netscape
        {
            if (document.layers)
            {merkeY=pageYOffset;}
        }
    }

    funktion setzeYPosition()
    {
        merkeYPosition();
        if (document.all)
        {document.all[idName].style.top=merkeY+posY;}
        else
        {
            if (document.layers)
            {document.layers[idName].top=merkeY+posY;}
        }
    }

    function merkeXPosition()
    {
        if (document.all)           // IE
        {merkeX=document.body.scrollLeft;}
        else                         // Netscape
        {
            if (document.layers)
            {merkeX=pageXOffset;}
        }
    }

```



[illegible]

Beispiel 2

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
//           feste Grafik, die nicht mitscrollt
//           Grafik steht immer rechts unten im Fenster
//           Die Grafik-Positionskorrektur-Geschwindigkeit ist leider abhängig von der
//           Scrollgeschwindigkeit,
//           so dass ein Springen der Grafik unvermeidlich ist,
//           wenn die Scrollgeschwindigkeit zu gross ist.

//*****
//
//           nachfolgende Variablen müssen vom Programmierer gesetzt werden
//
//*****
var GrafikBreiteLautIMGinBODY=30;           // identisch mit IMG-Werten laut BODY
var GrafikHoeheLautIMGinBODY =49;         // identisch mit IMG-Werten laut BODY
var GrafikAbstandZurFensterEckeRechtsUntenX=0; // in Pixel
var GrafikAbstandZurFensterEckeRechtsUntenY=0; // in Pixel
var GrafikPositionsKorrekturGeschwindigkeit=1; // >=1 wobei 1 = schnellste Positionskorrektur
```



```

//*****
//
//          nachfolgenden Code nicht verändern
//
//*****
//          Variablen hier ohne extra Funktion initialisieren
//          werden automatisch beim Dokumentladen belegt
//
//          BrowserTyp-Ermittlung
//          Annahme Netscape läuft
var BrowserTyp=true;
if (document.all)
{
    // IE läuft
    BrowserTyp=false;
}

// Browser-Fenster-Variablen
var BrowserFenster_AktuelleBreite    =0;
var BrowserFenster_AktuelleHoehe    =0;
var BrowserFenster_AktuelleScrollPositionX =0;
var BrowserFenster_AktuelleScrollPositionY =0;

// Grafik-Positions-Korrektur-Variablen
var GrafikPositionsKorrekturX=0;
var GrafikPositionsKorrekturY=0;

function GrafikPositionKorrigieren()
{
    // aktuelle Browserfenster-Daten ermitteln
    if (BrowserTyp)
    {
        // Netscape läuft
        BrowserFenster_AktuelleBreite    =window.innerWidth;
        BrowserFenster_AktuelleHoehe    =window.innerHeight;
        BrowserFenster_AktuelleScrollPositionX =window.pageXOffset;
        BrowserFenster_AktuelleScrollPositionY =window.pageYOffset;
    }
    else
    {
        //IE läuft
        BrowserFenster_AktuelleBreite    =document.body.clientWidth;
        BrowserFenster_AktuelleHoehe    =document.body.clientHeight;
        BrowserFenster_AktuelleScrollPositionX =document.body.scrollLeft;
        BrowserFenster_AktuelleScrollPositionY =document.body.scrollTop;
    }

    // Korrekturpositionen für Grafik ermitteln
    //          Achtung: Die Klammerung ist hier WICHTIG !!!
    GrafikPositionsKorrekturX=    BrowserFenster_AktuelleBreite
        - (GrafikBreiteLautIMGinBODY+GrafikAbstandZurFensterEckeRechtsUntenX)
        + BrowserFenster_AktuelleScrollPositionX;

    GrafikPositionsKorrekturY=    BrowserFenster_AktuelleHoehe
        - (GrafikHoeheLautIMGinBODY+GrafikAbstandZurFensterEckeRechtsUntenY)
        + BrowserFenster_AktuelleScrollPositionY;

    // Grafikposition korrigieren
    // eval () ermittelt String und liefert dessen Inhalt anstelle eval()
    // da der String ein StyleSheed-Kommando ist, wird das somit ausgeführt
    if (BrowserTyp)
    {
        // Netscape läuft
        eval("document.StyleSheedID.left=" + GrafikPositionsKorrekturX);
        eval("document.StyleSheedID.top=" + GrafikPositionsKorrekturY);
    }
    else
    {
        // IE läuft
        eval("StyleSheedID.style.pixelLeft=" + GrafikPositionsKorrekturX);
        eval("StyleSheedID.style.pixelTop=" + GrafikPositionsKorrekturY);
    }

    setTimeout("GrafikPositionKorrigieren()",GrafikPositionsKorrekturGeschwindigkeit);
}

```



```

}
//-->
</SCRIPT>
</HEAD>
<BODY onload="GrafikPositionKorrigieren();">
<DIV ID="StyleSheedID" STYLE="position:absolute; visibility:show; left:0px; top:0px; z-index:2">
  <CENTER>
    <IMG SRC="picture.gif" WIDTH="30" HEIGHT="49" BORDER="0">
  </CENTER>
</DIV>
Zeile<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
....
</BODY>
</HTML>

```

Beispiele für Bild verschwinden lassen:

Variante1: Bild ausserhalb des sichtbaren Bereiches positionieren

```

if (document.layers)
{
    document.layers[DIV_ID].left  = ( -1 * 40);
    document.layers[DIV_ID].top   = ( -1 * 20);
}
else
{
    if (document.all)
    {
        document.all.DIV_ID.style.left  = ( -1 * 40);
        document.all.DIV_ID_.style.top   = ( -1 * 20);
    }
}

<DIV ID="DIV_ID"
  STYLE="position:absolute;left=0px;top=0px;"
>
  <IMG NAME="IMG_ID"
    SCR="alt.jpg"
    HEIGHT=20
    WIDTH=40
    STYLE="...."
  >
</DIV>

```

Variante 2: Bild ersetzen durch transparentes 1x1 Pixel

```

document.IMG_ID.src="neu.gif";      // neu.gif ist transparent und 1x1 Pixel

<DIV ID="DIV_ID"
  STYLE="position:absolute;left=0px;top=0px;"
>
  <IMG NAME="IMG_ID"
    SCR="alt.jpg"
    HEIGHT=20
    WIDTH=40
    STYLE="...."
  >
</DIV>

```

Vairante 3: Sichtbarkeit verändern per style.visibility

Achtung: style.visibility=hidden lässt den Platz des Bildes im Layout **nicht** bestehen, was dann wichtig ist, wenn Umgebung des Bildes **nicht** per STYLE="position:absolute ..." erzeugt wurde.

Beispiel 1:

```

<HEAD>
<STYLE>
    .vis1 { visibility:visible }
    .vis2 { visibility:hidden }
</STYLE>

```



```

</HEAD>
<BODY>
  <IMG ID="ID_IMG" SRC="test.jpg">
  <P      onmouseover="ID_IMG.className='vis1'"
        onmouseout="ID_IMG.className='vis2'"
    >
        Testtext
  </P>
</BODY>

```

Beispiel 2:

```

<SCRIPT>
  function Verstecken()
  {ID_IMG.style.visibility="hidden"; }

  function Anzeigen()
  {ID_IMG.style.visibility="visible"; }
</SCRIPT>
<IMG ID="ID_IMG"
  SRC="test.jpeg">
<SPAN onmouseover="Verstecken()"
  onmouseout="Anzeigen()"
>
  Testtext
</SPAN>

```

Vairante 4: Sichtbarkeit verändern per style.display

Achtung: style.visibility=none lässt den Platz des Bildes im Layout bestehen, was dann wichtig ist, wenn Umgebung des Bildes **nicht** per STYLE="position:absolute ..." erzeugt wurde.

Beispiel 1:

```

<HEAD>
<STYLE>
  .vis1 { display:inline }
  .vis2 { display:none }
</STYLE>
</HEAD>
<BODY>
  <IMG ID="ID_IMG" SRC="test.jpg">
  <P      onmouseover="ID_IMG.className='vis1'"
        onmouseout="ID_IMG.className='vis2'"
    >
        Testtext
  </P>
</BODY>

```

Beispiel 2:

```

<SCRIPT>
  function Verstecken()
  {ID_IMG.style.display="none"; }

  function Anzeigen()
  {ID_IMG.style.display="inline"; }
</SCRIPT>
<IMG ID="ID_IMG"
  SRC="test.jpeg">
<SPAN onmouseover="Verstecken()"
  onmouseout="Anzeigen()"
>
  Testtext
</SPAN>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen



.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.border	Rahmendicke in Pixel
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.complete	Zustand des Ladens des Objektes
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.dynsrc	Adresse von Videoclip oder VRML
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.fileCreateDate	Datum der Dokumenterstellung
.fileModifiedDate	Datum der letzten Dokumentveränderung
.fileSize	Größe des Dokumentes
.fileUpdatedDate	Datum des letzten Datei-Updates
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.galleryImg	Toolbar "My Pictures Photo Support image toolbar" ein/ausschalten für aktuelles Image
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMap	server-seitige Image-Map
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.longDesc	Uniform Resource Identifier (URI) zur einer "long description" umwandeln
.loop	Anzahl der Wiederholungen nur Objekt BGSOUND bzw. IMG mit Sound bzw. INPUT-Element mit Sound
.lowsrc	Url des Images in geringerer Auflösung
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nameProp	Dateiname-Teil einer Url ohne Path und ohne Protokoll liefern
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem



	des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
.start	Start eines Videoclips oder VRML-Datei
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.useMap	Url oder Anker für client-seitige Image-Map
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern



	DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar
.click()	DOM wird geändert simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM- Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.detachEvent()	DOM nicht geändert Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_ bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.hasChildNodes()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten



	(Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endtag geparkt DOM wird geändert

4.3.2.2.4.3.23. document.images Collection des Internet Explorer

Feld der Zeiger aller img Objekte

Elementefolge laut HTML-Kodung

Achtung: Per new Image() erzeugte Bilder sind **nicht** Bestandteil der Collection !!



Diese Collection umfasst **nicht** das Objekt `input.image`, da für dieses Objekt die `childrenCollection` verwendet werden muss !

Syntax:

```
[ var ZeigerAufFeld = ] document.images
[ var ZeigerAufFeldElement = ] document.images [Index [, SubIndex] ]
```

Index	Integer ab 0
oder	String Name oder ID des Elementes
	muss in [] kodiert sein
SubIndex	optional
	nur kodieren wenn Index ein String ist
	Integer als Unterindex also Unterelement eines Elementes

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

`.item()` Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit `<INPUT TYPE=image ...>` da dafür die `children-Collection` verwendet werden muss !!!

`.namedItem()` Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

`.tags()` Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe `tags` Collection des DOM

`.urns()` Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.24. input Objekt und seine Varianten

Varianten des Input Controls (Eingabe-Steuerelementes) z.B. im Formular

festlegbar per TYPE-Attribut:

- button
- checkbox
- file
- hidden
- image
- password
- radio
- reset
- submit
- text

wobei der Wert des TYPE-Attributes **wahlweise**

in " " bzw. ' ' oder ohne " " bzw. ' '

kodiert werden kann.

Das input Objekt ist

Prototyp für das Objekt `document.form.input`
nutzt weitere Objekte wie `button` und `img`.

Sämtliche Input-Varianten besitzen das ID-Attribut, über das der Script-Zugriff möglich ist.

Nachfolgende Syntax-Beschreibungen zeigen

die HTML-Verwendung der Input-Varianten, wobei aus Gründen der Übersichtlichkeit auf die Kodierung des ID-Attributes verzichtet

wurde.

Eigenschaften und Methoden **nur** beim Internet Explorers.

Erzeugung in HTML:

Beispiel Alle Elemente im Formular müssen **NAME**-Attribut erhalten, wenn die Daten der Elemente gesendet werden sollen !

```
<FORM ACTION="http://intranet/test" METHOD=POST>
  <INPUT TYPE="TEXT" NAME="CONTROL1" VALUE="Ihr Name">
  <BR>
  <P>Password</P>
  <INPUT TYPE="PASSWORD" NAME="CONTROL2">
  <P>Farbe-Auswahl</P>
  <BR>
  <INPUT TYPE="RADIO" NAME="CONTROL3" VALUE="0" CHECKED>Rot
  <INPUT TYPE="RADIO" NAME="CONTROL3" VALUE="1">Gelb
  <INPUT TYPE="RADIO" NAME="CONTROL3" VALUE="2">Blau
  <P>Ihr Kommentar</P>
  <BR>
  <INPUT TYPE="TEXT" NAME="CONTROL4" SIZE="20,5" MAXLENGTH="250">
  <BR>
  <INPUT TYPE="CHECKBOX" NAME="CONTROL5" CHECKED>Senden
```



```

<BR>
<INPUT TYPE="SUBMIT" VALUE="OK">
<INPUT TYPE="RESET" VALUE="Reset">
</FORM>

```

Eigenschaften beim Internet Explorer:

Diese Eigenschaften sind nicht für alle INPUT-Varianten sinnvoll anwendbar

Bsp.: Loop für ein Image-Map

.accept	Liste von Multipurpose Internet Mail Extensions (MIME)-Typen
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
.complete	Zustand des Ladens des Objektes
.dynsrc	Adresse von Videoclip oder VRML
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.lowsrc	Url des Images in geringerer Auflösung
.start	Start eines Videoclips oder VRML-Datei
.useMap	Url oder Anker für client-seitige Image-Map
.value	Wert eines Objekt-Attributes
	Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.vspace	vertikaler Abstand in Pixel zum Elternobjekt

Methoden beim Internet Explorer:

keine

4.3.2.2.4.3.24.1. input button Objekt

Taste-Control-Element (Button-Control-Element)

Hinweis: Es gibt noch das button Objekt (siehe dort)

Erzeugung in HTML:

<INPUT TYPE=button> auch als String "button" kodierbar

Beispiel

```

<INPUT TYPE=button
VALUE="betaetigen!"
NAME="button1"
onClick="alert('Das input button Objekt in Aktion')"
>

```

Eigenschaften beim Internet Explorer:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst
ATOMICSELECTION	vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.begin	Selektierbarkeit des Objektes einstellen Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2



.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden <code>var Zeiger = eval(object.id);</code>
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes



Anspringen verbunden mit Focus erhalten
 --> Ereignisse werden ausgelöst !!
 unter IE 5.x
 onblur, onfocus
 ab IE 5.x
 onblur, onfocus, onkeydown, onkeypress, onkeyup

Anspringen default per TAB-Taste
 für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT,
 SELECT, TEXTAREA

Anspringen default nicht per TAB-Taste
 für APPLET, DIV, FRAMESET, SPAN, TABLE, TD

.tagName Tag-Bezeichner des Objektes
 .tagUrn Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
 .timeContainer Typ der Timeline des Objektes
 siehe Objekt currTimeState und Behavior .style.time2
 .title Tooltip-Text bei Mouse over über Objekt
 .type Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
 .uniqueID durch den Browser automatisch-generiertes ID des Objektes
 Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
 kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
 UNSELECTABLE Selektionsfähigkeit eines Objektes
 .value Wert eines Objekt-Attributes
 Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
 .width Breite des Objektes in Pixel

Methoden beim Internet Explorer:

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen
 Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst
 werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
 ab IE 5.x bis unter IE 5.5
 .appendChild() Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
 DOM wird geändert
 Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
 Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag
 (falls vorhanden) des Knoten geparkt wurde
 .applyElement() Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz
 laut DOM liefern
 DOM wird geändert
 Element kann selbst Kinder haben
 Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
 Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im
 im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
 .attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
 NICHT verkettet sondern in **Zufallsfolge**, es sei denn
 die Handler prüfen ihre Aufruffolge (muss programmiert werden)
 .blur() Element den Focus wegnehmen und Event onblur auslösen
 Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
 vor IE 5.0 TABINDEX-Attribut muss kodiert sein
 ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
 .clearAttributes() alle HTML-Attribute eines Objektes entfernen
 außer ID, STYLE und per Script definierte Attribute
 Script-erzeugte Attribute nicht entfernbar
 DOM wird geändert
 .click() simuliert einen Klick auf das Element und löst onclick-Event aus
 manipuliert nicht den Focus
 .cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
 DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-
 Objektes im Hauptspeicher außerhalb des DOM)
 .componentFromPoint() Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
 auch für CSS-Layout
 onmouseover-Event hat **nicht die Pixelgenauigkeit** wie die Angaben der Methode .componentFromPoint()
 also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos
 erreicht haben
 Overbereich der Maus ist mehr als 1 Pixel gross
 beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
 .contains() prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern
 (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
 DOM nicht geändert
 .createTextRange() Textbereich erzeugen
 .detachEvent() Abschalten des Registrieren eines Events durch Eventhandler
 wobei Registrierung mit Methode .attachEvent() aktiviert wurde
 Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner



	zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern



	Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
	DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein
	DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endtags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode <code>.focus()</code> verwenden)
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
	ab IE 5.5
	Hinweis: ausschalten per Methode <code>.releaseCapture()</code>
.setExpression()	Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt
	DOM wird geändert

4.3.2.2.4.3.24.2. input checkbox Objekt

Checkbox-Control

Standardwert ist "on"

Style-Wert zu height und width ab IE 5.x (innere Höhe bzw. Breite): wenn height bzw. width

>= 20 Pixel

Dicke Padding um die Checkbox	4 Pixel
innere Höhe bzw innere Breite	8 Pixel

< 20 Pixel **und** > 13 Pixel

Dicke Padding um die Checkbox	(height -13) / 2 Pixel bzw. (width -13) / 2 Pixel
innere Höhe bzw innere Breite	unverändert übernommen

< 13 Pixel

Dicke Padding um die Checkbox	0 Pixel
innere Höhe bzw innere Breite	unverändert übernommen

Erzeugung in HTML:

<INPUT TYPE=checkbox> auch als String "checkbox" kodierbar

Beispiel

```

<SCRIPT>
function Anzeige1()
{ID_Span.insertAdjacentHTML("Checkbox 1 aktiv");}

function Anzeige2()
{ID_Span.insertAdjacentHTML("Checkbox 2 aktiv");}
</SCRIPT>
<INPUT TYPE=checkbox
NAME="checkbox1"
CHECKED
onclick="Anzeige1()"

```



```

>Aktion 1
<INPUT TYPE=checkbox
NAME="checkbox2"
onclick="Anzeige2()"
>Aktion 2
<SPAN ID="ID_Span">
Test-Text
</SPAN>

```

Eigenschaften beim Internet Explorer:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.checked	Selektionsstatus des Checkbox-Control bzw. Radio Button-Control bezüglich Selektion durch User
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultChecked	Selektionsstatus des Checkbox-Control bzw. Radio Button-Control bzw. Radio Button-Control bezüglich Standard-Selektion
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.indeterminate	Grauzustand (Dimmed) und Selektiertheit des Checkbox-Control
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)



.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
.status	Selektionsstatus eines Control-Elementes Control-Element: kann durch User interaktiv verändert werden
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes



.width	Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression()



	zu definieren
	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
	DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
	DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich
	HTML- und Script-Code müssen syntaktisch korrekt sein
	wenn nicht, so wird das Einfügen nicht ausgeführt
	eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist
	bei Script-Code: <SCRIPT DEFER> muss kodiert werden
	DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
	DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein
	Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
	Attribute sind: HTML
	Events
	Styles
	ab IE 5.01 auch ID, NAME
	Achtung: Diese Methode ist mir Vorsicht zu genießen !!
	DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur
	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt
	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes
	Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
	per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
	DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
	DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
	DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern
	Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
	DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient.
	Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
	DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt
	ersetzende Objekt muss per Methode .createElement() erzeugt worden sein
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
	sichtbar erst mit parsen des Endetags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren
	setzt nicht den Focus (dafür Methode .focus() verwenden)
.setActive()	Objekt für die Eventdurchführung aktivieren
	aber ohne es zu fokussieren



und ohne es scrollbar zu machen

`.setAttribute()` Wert von vorhandenem Attribut setzen
wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt

`.setAttributeNode()` DOM wird nur bei Erzeugung geändert
Attribut einem Knoten zuweisen und Referenz liefern
DOM wird geändert

`.setCapture()` Maus-Überwachung einschalten für ein Objekt
Maus-Events sind : `onmousedown`, `onmouseup`, `onmousemove`, `onclick`, `ondblclick`,
`onmouseover` und `onmouseout`.
ab IE 5.5

`.setExpression()` Hinweis: ausschalten per Methode `.releaseCapture()`
Wert definieren, der als Ausdruck für die Methode `.getExpression()` zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
`objekt.style.eigenschaft.`
dient
Ausdruck nur als Script kodierbar
DOM wird nicht geändert

`.swapNode()` Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
nur sichtbar wenn Endetag geparkt
DOM wird geändert

4.3.2.2.4.3.24.3. *input file Objekt*

File Upload-Control mit Dialogbox

ab IE 4.x

folgende Voraussetzungen **müssen** erfüllt werden:

Control muss in einem Formular liegen

NAME-Attribut muss kodiert sein

Formular mit Attribut `METHOD` auf "post" gesetzt

`ENCTYPE` auf "multipart/form-data"

Server muss den Enctype "multipart/form-data" verarbeiten können (eventuell CGI- oder ASP-Script dort nötig)

Erzeugung in HTML:

Beispiel mit Serverscript per ASP:

HTML-Dokument:

```
<FORM NAME="Formular"
ACTION="script.asp"
ENCTYPE="multipart/form-data"
METHOD="post"
>
<INPUT TYPE="file" NAME="testdatei">
<INPUT TYPE="submit" VALUE="Upload der Datei">
</FORM>
```

script.asp:

```
<% @ LANGUAGE = JScript %>
<%
    Response.buffer=true;
%>
<HTML>
<BODY>
    <H1>Upload Status</H1>
    <P>
        Zielort: <% Response.Write(Request.Form("TargetURL")) %>
    </P>
    <%
        Response.write("<P>Dateiname: " + Request.Form("FileName") + "</P>");
        Response.write("<P>Dateigrösse: " + Request.Form("FileSize") + "</P>");
        Response.write("<P>Dateipfad: " + Request.Form("FilePath") +
"</P>");
    %>
</BODY>
</HTML>
```

Eigenschaften beim Internet Explorer:

`.accessKey` Tastaturzugriff auf ein Objekt per Alt + Taste
bei Ausführung der Tastenkombination wird
das Sprungziel wird focussiert
die Sprungquelle defocussiert
das Focus-Ereignis ausgelöst
vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt

ATOMICSELECTION
`.begin` Selektierbarkeit des Objektes einstellen
Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft `.timeAction`
siehe Objekt `currTimeState` und `Behavior.style.time2`

`.canHaveChildren` prüfen ob Kind möglich ist, also ob Objekt Parent sein kann

`.canHaveHTML` prüfen ob Objekt HTML-Tags enthalten darf



.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes



	nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberem sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style)
	Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit).
	nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior)
	ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !
	siehe Objekt currTimeState und Behavior .style.time2
	siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
	für Anspringen des Dokumentes
	Anspringen verbunden mit Focus erhalten
	--> Ereignisse werden ausgelöst !!
	unter IE 5.x
	onblur, onfocus
	ab IE 5.x
	onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste
	für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste
	für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes
	siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes
	Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
	DOM wird geändert
	Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
	Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
	vor IE 5.0 TABINDEX-Attribut muss kodiert sein
	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen
	außer ID, STYLE und per Script definierte Attribute



	Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME



	Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur
	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
	DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
	DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
	DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
	DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren
	setzt nicht den Focus (dafür Methode .focus() verwenden)
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
	Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt
	DOM wird geändert

4.3.2.2.4.3.24.4. **input hidden Objekt**

ist **kein** Control-Element

Ersatz für Cookies: Übertragung von für den User nicht sichtbarer Daten zum Server
gesendet wird der Wert

zu sendender Wert durch Programmierer festlegbar und nicht durch User beeinflussbar

Achtung: Dieses Objekt kann missbräuchlich benutzt werden !

Erzeugung in HTML:

<INPUT TYPE=hidden ...> auch als "hidden" kodierbar



Eigenschaften beim Internet Explorer:

ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)



.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar

Methoden beim Internet Explorer:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbare DOM wird geändert
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.createTextRange()	Textbereich erzeugen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert



<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mit Vorsicht zu genießen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5 Hinweis: ausschalten per Methode <code>.releaseCapture()</code>
<code>.setExpression()</code>	Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.24.5. input image Objekt

Image-Control für Senden eines Formulars durch Klick auf das Bild (interaktives Bild **nur** zum Zweck des Submit) gesendet werden **nur** die Koordinaten der linken oberen Ecke (in Pixel) gesendet
wobei es sich darin um interne Eigenschaften handelt, die nicht manipulierbar sind

`.x` für Spalte
`.y` für Zeile

ab IE 5.0 werden folgende Image-Medien unterstützt:

<code>.avi</code>	Audio-Visual Interleaved (AVI)
<code>.bmp</code>	Windows Bitmap (BMP)
<code>.emf</code>	Windows Enhanced Metafile (EMF)
<code>.gif</code>	Graphics Interchange Format (GIF)
<code>.jpg, .jpeg</code>	Joint Photographic Experts Group (JPEG)
<code>.mov</code>	Apple QuickTime Movie (MOV)
<code>.mpg, .mpeg</code>	Motion Picture Experts Group (MPEG)



.png	Portable Network Graphics (PNG)
.wmf	Windows Metafile (WMF)
.xbm	X Bitmap (XBM)

basiert auf dem img Objekt

Erzeugung in HTML:

<INPUT TYPE=image> auch als "image" kodierbar

Eigenschaften beim Internet Explorer:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.complete	Zustand des Ladens des Objektes
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.dynsrc	Adresse von Videoclip oder VRML
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.loop	Anzahl der Wiederholungen nur Objekt BGSOUND bzw. IMG mit Sound bzw. INPUT-Element mit Sound
.lowsrc	Url des Images in geringerer Auflösung
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker



.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
.start	Start eines Videoclips oder VRML-Datei
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde



UNSELECTABLE	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.value	Selektionsfähigkeit eines Objektes Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle



<code>.getExpression()</code>	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
<code>.hasChildNodes()</code>	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
<code>.removeAttribute()</code>	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.select()</code>	Bereich des Input-Objektes im Formular markieren



.setActive()	setzt nicht den Focus (dafür Methode .focus() verwenden) Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.24.6. document.images Collection des Internet Explorer

Feld der Zeiger aller img Objekte

Elementefolge laut HTML-Koding

Achtung: Per new Image() erzeugte Bilder sind **nicht** Bestandteil der Collection !!

Diese Collection umfasst **nicht** das Objekt input.image, da für dieses Objekt die childrenCollection verwendet werden muss !

Syntax:

```
[ var ZeigerAufFeld = ] document.images
[ var ZeigerAufFeldElement = ] document.images [Index [, SubIndex] ]
```

Index	Integer ab 0
oder	String Name oder ID des Elementes muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.tags()	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.24.7. input password Objekt

Text-Control aus einer Zeile mit Dummy-Anzeige während der Eingabe der Zeichen durch User

Erzeugung in HTML:

<INPUT TYPE=password> auch als "password" kodierbar

Beispiel <SCRIPT>

```
function Pruefen()
{
    if (ID_InputButton.value == "Otto")
    {
        if (ID_InputPassword.value == "Waalkes")
        {alert("Password korrekt");}
    }
}
```

</SCRIPT>

Username <INPUT TYPE=button ID="ID_InputButton" onclick="Pruefen()">
Password <INPUT TYPE=password ID="ID_InputPassword" onclick=Pruefen()">

Eigenschaften beim Internet Explorer:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste



	bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.autocomplete	Status des Autovervollständigung zum Formular
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
.isContentEditable	Zeiger aus ID bilden var Zeiger = eval(object.id); Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.maxLength	Maximale Anzahl der durch User eingebaren Zeichen in einem Text-Control
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x



.outerHTML	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readOnly	Editierbarkeit eines Text-Controls Editierung bedeutet Focuserhalt !
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.vcard_name	Werte für vCard für Autocomplete (Autovervollständigung) bei Formular per Text-Control
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert



	<p>Zeiger wird zugleich immer am Ende der Collection childNodes angehängen</p> <p>Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde</p>
.applyElement()	<p>Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern</p> <p>DOM wird geändert</p> <p>Element kann selbst Kinder haben</p> <p>Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde</p> <p>Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !</p>
.attachEvent()	<p>Einschalten des Registrieren eines Events durch Eventhandler</p> <p>Hinweis: Abschalten mit Methode .detachEvent()</p> <p>Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)</p>
.blur()	<p>Element den Focus wegnehmen und Event onblur auslösen</p> <p>Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !</p> <p>vor IE 5.0 TABINDEX-Attribut muss kodiert sein</p> <p>ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein</p>
.clearAttributes()	<p>alle HTML-Attribute eines Objektes entfernen</p> <p>außer ID, STYLE und per Script definierte Attribute</p> <p>Script-erzeugte Attribute nicht entfernbare</p> <p>DOM wird geändert</p>
.click()	<p>simuliert einen Klick auf das Element und löst onclick-Event aus</p> <p>manipuliert nicht den Focus</p>
.cloneNode()	<p>Objekt klonen und Referenz des erzeugten Klone liefern</p> <p>DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)</p>
.componentFromPoint()	<p>Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt</p> <p>auch für CSS-Layout</p> <p>onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben</p> <p>Overbereich der Maus ist mehr als 1 Pixel gross</p> <p>beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.</p>
.contains()	<p>prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist</p> <p>DOM nicht geändert</p>
.createTextRange()	Textbereich erzeugen
.detachEvent()	<p>Abschalten des Registrieren eines Events durch Eventhandler</p> <p>wobei Registrierung mit Methode .attachEvent() aktiviert wurde</p> <p>Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)</p>
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	<p>Focus setzen und Focus-Event auslösen</p> <p>nur nach dem kompletten Laden des Dokumentes</p> <p>vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen</p>
.getAdjacentText()	<p>Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann</p> <p>Text kann HTML-Tags enthalten, muss aber nicht</p> <p>DOM nicht geändert</p>
.getAttribute()	<p>Wert eines per HTML erzeugten Attributes liefern</p> <p>DOM nicht geändert</p>
.getAttributeNode()	<p>Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft.</p> <p>Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht</p> <p>Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt</p> <p>Wert des Attributes wird somit über die Referenz laut DOM erreichbar</p> <p>DOM nicht geändert</p>
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	<p>Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster</p> <p>Feld mit Index als Integer ab 0</p> <p>pro Eintrag ein Rectangle</p>
.getExpression()	<p>Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern</p> <p>Style-Eigenschaft ist per Methoden</p> <p>expression() oder setExpression()</p> <p>zu definieren</p> <p>DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)</p>
.hasChildNodes()	<p>prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt</p> <p>DOM nicht geändert</p>
.insertAdjacentElement()	<p>Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann</p> <p>wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM</p>



	nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode .focus() verwenden)
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,



onmouseover und onmouseout.
 ab IE 5.5
 Hinweis: ausschalten per Methode .releaseCapture()
 .setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 .swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.24.8. input radio Objekt

Radio-Button-Control z.B. für Formular
 ermöglicht Auswahl (Selektion) genau eines Eintrages aus Menge von Einträgen:

jeder Eintrag ist Radio-Button-Control
 kann nur selektiert werden wenn NAME-Attribut kodiert wurde
 zu jedem Zeitpunkt kann nur ein Eintrag ausgewählt sein

Menge von Einträgen ist Gruppe:
 alle Elemente der Gruppe haben identischen Wert im NAME-Attribut
 mehrere Gruppen zulässig mit je eigenem Namen
 Standard-Element markierbar per Eigenschaft .checked (siehe dort)

senden beim Formular: nur das ausgewählte (selektierte) Element der Gruppe:
 gesendet wird der Wert und NAME-Attribut-Wert

Erzeugung in HTML:

Beispiel

```
<SCRIPT>
function Anzeigen()
{
    if (radio[0].checked)
    {alert("Eintrag 1");}
    else
    {
        if (radio[1].checked)
        {alert("Eintrag 2");}
        else
        {alert("Eintrag 3");}
    }
}
</SCRIPT>
Gruppe <B>GemeinsamerName<B>
<BR>
<INPUT TYPE=radio NAME="GemeinsamerName" CHECKED>Eintrag 1
<INPUT TYPE=radio NAME="GemeinsamerName">Eintrag 2
<INPUT TYPE=radio NAME="GemeinsamerName">Eintrag 3
<INPUT TYPE=button
VALUE="Anzeige des aktuell selektierten Eintrages"
onClick="Anzeigen()"
>
```

Eigenschaften beim Internet Explorer:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste
 bei Ausführung der Tastenkombination wird
 das Sprungziel wird focussiert
 die Sprungquelle defocussiert
 das Focus-Ereignis ausgelöst
 vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
 ATOMICSELECTION Selektierbarkeit des Objektes einstellen
 .begin Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
 siehe Objekt currTimeState und Behavior .style.time2
 .canHaveChildren prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
 .canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf
 .checked Selektionsstatus des Checkbox-Control bzw. Radio Button-Control
 bezüglich Selektion durch User
 .className Klassenreferenz, Klassenname
 .clientHeight Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
 ohne Rahmen
 ohne Scrollbalken
 .clientLeft Abstand in Pixel zum linken Rand des Fensters
 .clientTop Abstand in Pixel zum oberen Rand des Fensters



.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultChecked	Selektionsstatus des Checkbox-Control bzw. Radio Button-Control bzw. Radio Button-Control bezüglich Standard-Selektion
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes



.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
.status	Selektionsstatus eines Control-Elementes Control-Element: kann durch User interaktiv verändert werden
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !



	vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen



	Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode .focus() verwenden)
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.24.9. input reset Objekt

Reset-Button für Formular-Reset: Standardgemäß alle Formular-Elemente auf Initialwert setzen.
VALUE-Attribut für Titel (Label) des Reset-Button verwenden
Wert wird nicht gesendet



Erzeugung in HTML:

<INPUT TYPE=reset> auch als "reset" kodierbar

Eigenschaften beim Internet Explorer:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem



	des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x
	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag
	nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag



	(falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.createTextRange()	Textbereich erzeugen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert



<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.select()</code>	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode <code>.focus()</code> verwenden)
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5



Hinweis: ausschalten per Methode .releaseCapture()
 Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.24.10. input submit Objekt

Submit-Button für Formular

Label im VALUE-Attribut kodieren (Standardtext ist "Submit Query")

Senden nur möglich, wenn NAME-Attribut kodiert wurde:

gesendet wird Wert von NAME- und VALUE-Attribut

Erzeugung in HTML:

<INPUT TYPE=submit> auch als "submit" kodierbar

Eigenschaften beim Internet Explorer:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste
 bei Ausführung der Tastenkombination wird
 das Sprungziel wird focussiert
 die Sprungquelle defocussiert
 das Focus-Ereignis ausgelöst
 vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
 ATOMICSELECTION
 .begin Selektierbarkeit des Objektes einstellen
 Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
 siehe Objekt currTimeState und Behavior .style.time2
 .canHaveChildren prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
 .canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf
 .className Klassenreferenz, Klassenname
 .clientHeight Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
 ohne Rahmen
 ohne Scrollbalken
 .clientLeft Abstand in Pixel zum linken Rand des Fensters
 .clientTop Abstand in Pixel zum oberen Rand des Fensters
 .clientWidth Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
 ohne Rahmen
 ohne Scrollbalken
 .contentEditable Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat)
 Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
 .defaultValue Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
 .dir Umflussrichtung
 .disabled Interaktionsfähigkeit
 nur wenn sichtbar so User-Interaktion möglich
 .end Objektaktivitäten laut Eigenschaft .timeAction beenden
 ab IE 6.x
 alternativ: Eigenschaft .dur
 siehe Objekt currTimeState und Behavior .style.time2
 .firstChild Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
 .form Zeiger auf das Formular (Formular als Container)
 ab IE 6.x für Elemente fieldSet, label, legend
 .hasMedia Objekt ist HTML-Media-Objekt
 siehe Objekt currTimeState und Behavior .style.time2
 .hideFocus Focussierbarkeit
 .id Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
 Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt
 und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und
 betroffenes Objekt die Eigenschaft .uniqueID kennen).
 Zeiger aus ID bilden var Zeiger = eval(object.id);
 .isContentEditable Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
 Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
 .isDisabled Interaktionsfähigkeit
 nur wenn sichtbar so User-Interaktion möglich
 .isMultiLine Mehrzeiligkeit des Objektinhaltes
 .isTextEdit Erzeugbarkeit eines Textbereiches
 .lang Sprache für Anzeige von Sonderzeichen etc.
 .language Sprache für Script festlegen
 .lastChild Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
 .name Name des Objektes (nicht ID !!!)
 muss beim Formular für alle zu sendenden Felder kodiert sein !!
 Element darf nicht per Methode .createElement() erzeugt worden sein
 .nextSibling Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes



.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currentTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currentTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde



UNSELECTABLE	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.value	Selektionsfähigkeit eines Objektes Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.width	Breite des Objektes in Pixel
Methoden beim Internet Explorer:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.createTextRange()	Textbereich erzeugen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnet zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0



	pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein



<code>.select()</code>	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode <code>.focus()</code> verwenden)
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
<code>.setAttributeNode()</code>	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
<code>.setCapture()</code>	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5 Hinweis: ausschalten per Methode <code>.releaseCapture()</code>
<code>.setExpression()</code>	Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> . dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.24.11. *input text Objekt*

Einzeilige Textbox, in der User eingeben kann

Breite der Textbox in Zeichen im `SIZE`-Attribut kodieren.

Maximale Anzahl der eingebbaren Zeichen in Attribut `MAXLENGTH`s kodieren.

Wenn `SIZE < MAXLENGTH` so wird automatisch horizontaler Scrollbalken erzeugt.

Erzeugung in HTML:

Beispiel

```
<SCRIPT>
    function Anzeige()
    {alert("Es wurde eingegeben" + textbox.value); }
</SCRIPT>

<INPUT TYPE=text VALUE="" NAME="textbox" SIZE=15 onclick="Anzeige()">
```

Eigenschaften beim Internet Explorer:

<code>.accessKey</code>	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst
<code>ATOMICSELECTION</code>	vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt Selektierbarkeit des Objektes einstellen
<code>.autocomplete</code>	Status des Autovervollständigung zum Formular
<code>.begin</code>	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft <code>.timeAction</code> siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code>
<code>.canHaveChildren</code>	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
<code>.canHaveHTML</code>	prüfen ob Objekt HTML-Tags enthalten darf
<code>.className</code>	Klassenreferenz, Klassenname
<code>.clientHeight</code>	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
<code>.clientLeft</code>	Abstand in Pixel zum linken Rand des Fensters
<code>.clientTop</code>	Abstand in Pixel zum oberen Rand des Fensters
<code>.clientWidth</code>	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
<code>.contentEditable</code>	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
<code>.dataFld</code>	Datenquelle-Name vergeben (ID)
<code>.dataSrc</code>	Datenquelle als Anker festlegen
<code>.defaultValue</code>	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars
<code>.dir</code>	Umflussrichtung
<code>.disabled</code>	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
<code>.end</code>	Objektaktivitäten laut Eigenschaft <code>.timeAction</code> beenden ab IE 6.x alternativ: Eigenschaft <code>.dur</code> siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code>
<code>.firstChild</code>	Zeiger auf das ERSTE Kind laut <code>childNodes-Collection</code> eines Objektes



.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.maxLength	Maximale Anzahl der durch User eingebaren Zeichen in einem Text-Control
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readOnly	Editierbarkeit eines Text-Controls Editierung bedeutet Focuserhalt ! aktueller Status des Objektes beim Füllen des Objektes mit Daten
.readyState	
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.size	Dimension eines Input-Objektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !



	siehe Objekt currTimeState und Behavior .style.time2
	siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes
	siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT)
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.vcard_name	Werte für vCard für Autocomplete (Autovervollständigung) bei Formular per Text-Control
.width	Breite des Objektes in Pixel
<u>Methoden beim Internet Explorer:</u>	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM- Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben



	Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.createTextRange()	DOM nicht geändert Textbereich erzeugen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.insertAdjacentHTML()	DOM wird geändert HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.insertBefore()	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur
.releaseCapture()	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen Maus-Überwachung ausschalten für ein Objekt Maus-Events sind: onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst



.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Bereich des Input-Objektes im Formular markieren setzt nicht den Focus (dafür Methode <code>.focus()</code> verwenden)
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setCapture()	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
.setExpression()	ab IE 5.5 Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.25. label Objekt

Aufschrift/Beschriftung (Etikett) eines Elementes der Webseite. Nicht verwechseln mit Titel, da Titel in HTML anders verwendet wird.

Label sind nicht verschachtelbar.

Input-Elemente (Objekt `input xxxx`) haben z.T. eigenen Labelkodierung im Attribut `VALUE` und benötigen das Objekt `label` nicht.

Erzeugung in HTML:

Kodierung der Beschriftung:

Dieses Objekt verlangt eine Kodierung des Attributes `ID` im Label **und** im zu beschriftenden Element. Das `ID` ist das Bindeglied zur Realisierung einer Beschriftung. Es ist nicht das `ID`-Attribut !!!

```
Bsp.: <LABEL FOR="ID_Label">Beschriftung</LABEL>
      <INPUT TYPE="text"
            ID="ID_Label"
            VALUE="Textbox mit Beschriftung"
            SIZE="20"
      >
```

Elementzugriff per Tastaturkürzel:

Label wird vorrangig zur Realisierung des Zugriffes auf das Element per Tastaturkürzel verwendet:
im Label das Attribut `ACCESSKEY` kodieren

```
Bsp.: <LABEL FOR="ID_Label" ACCESSKEY="I">
```



Zugrifferklärung z.B. per mit Unterstreichung eines Buchstabens als Tastaturkürzel
per STYLE="text-decoration: underline"

Bsp.: 1: Press Alt+1 für Anwahl des Elementes

Beispiel

```
<LABEL FOR="ID_Label" ACCESSKEY="1">
  <SPAN STYLE="text-decoration:underline;">1</SPAN>: Press Alt+1 für Anwahl der Textbox
</LABEL>
<INPUT TYPE="text" ID="ID_Label" NAME="TXT1" VALUE="Textbox" SIZE="20" TABINDEX="1">
```

NAME-Attribut nur nötig bei Formular
 dient nur der Erklärung des Tastaturkürzel und der Unterstreichung des Kürzelzeichens

Hinweis: Im Formular müssen alle Elemente - wie das Formular selbst - das Attribut NAME erhalten.

Mausklick auf Label erzeugt das Event onclick, bewirkt aber keinen Aufruf per Tastaturkürzel (es sei denn, es wird manuell als Ersatz von ACCESSKEY programmiert).

LABEL Objekt des Internet Explorer:

ab IE 4.x

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hideFocus	Focussierbarkeit
.htmlFor	ID des Label nur für die Realisierung der Beschriftung ist nicht das ID laut ID-Attribut per label Objekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit



	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhalts
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
----------------	---



	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementByName()



.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.insertBefore()	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein



.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.26. link Objekt

siehe auch Objektes a

Erzeugung unter HTML:

```
<LINK
    HREF="url"                                // protocol://[host_name[:port]/]dateiname
                                              // protocol://[host_name[:port]/]dateiname#hashtext
                                              // protocol://[host_name[:port]/]dateiname?serachtext
                                              // Bsp: http://www.test.com:8888/test.html
    NAME="anker_name_ohne_#"                // ist nicht der logische link_name !!
                                              // Netscape + Internet Explorer
oder    ID="anker_name_ohne_#"              // ist der logische link_name !!
                                              // nur Internet Explorer ab 4.x
    TARGET="logischer_window_name"
    onClick="eventhandler1"
    onDbClick="eventhandler2"
    onMouseOut="eventhandler3"
    onMouseOver="eventhandler4"
    onMouseDown="eventhandler5"
    onMouseUp="eventhandler6"
>
    link_text_immer_schreiben
</LINK>
```

Zugriff:

```
document.links[index]

index:    ab 0
          Nummer des Links im Dokument
          Anzahl aller Links lautet document.links.length
```

Beispiel für globalen Style per HEAD:

```
<HEAD>
<STYLE>
    .an {text-decoration: underline overline; color:blue;}
    .aus {text-decoration: none; color:black;}
</STYLE>
</HEAD>
<BODY>
    <A      HREF="test.htm"
            CLASS="aus"
            onmouseover="this.className='an';"
            onmouseout="this.className='aus';"
    >
    </A>
</BODY>
```

Eigenschaften (ausgewählte):

.className	Klassenreferenz
.hash	entspricht #hashtext zum Anspringen eines Ankers hier den Ankernamen mit vorgesetztem # ablegen
.host	entspricht hostname:port
.hostname	entspricht nur hostname
.href	gesamter Url (kompletter Url)



	zum Anspringen eines Ankers
	hier den Ankernamen ohne vorgesetztem # ablegen
.id	entspricht ID
	nur IE ab 4.x
.innerText	ist der link_text
	nur IE ab 4.x
	lesen und schreiben
.name	entspricht NAME
.offsetHeight	Pixelhöhe
	nur IE ab 4.x
.offsetLeft	Pixelpos bezüglich linken Rand des HTML-Dokumentes
	nur IE ab 4.x
.offsetTop	Pixelpos bezüglich oberen Rand des HTML-Dokumentes
	nur IE ab 4.x
.offsetWidth	Pixelbreite
	nur IE ab 4.x
.offsetLeft	Pixelpos bezüglich linken Rand des HTML-Dokumentes
.pathname	entspricht aus url /dateiname
	bzw. /dateiname#hashtext
	bzw. /dateiname?searchtext
	lesen und schreiben
.port	entspricht port; lesen und schreiben
.protocol	entspricht Protokoll mit Doppelpunkt z.B. "http:"
	lesen und schreiben
.search	entspricht ?searchtext
	lesen und schreiben
.style	Zeiger auf das style-Objekt
	nur IE ab 4.x
.tagName	enthält "A"
	nur IE ab 4.x
.target	entspricht TARGET
	kann auch sein
	_blank
	_parent
	_search
	_self
	_top
	lesen und schreiben
.text	enthält den link_text
	nur lesen
	nur NS ab 4.x
.x	horizontale Pixelpos gegenüber linke, obere Ecke (0,0) des HTML-Dokumentes
	nur lesen
	nur NS ab 4.x
.y	vertikale Pixelpos gegenüber linke, obere Ecke (0,0) des HTML-Dokumentes
	nur lesen
	nur NS ab 4.x
<u>Methoden (ausgewählte):</u>	
.attachEvent()	siehe Eventbehandlung des IE ab 5.x
	nur IE ab 5.x
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
	vor IE 5.0 TABINDEX-Attribut muss kodiert sein
	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.detachEvent()	siehe Eventbehandlung des IE ab 5.x
	nur IE ab 5.x
.focus()	Focus setzen und Focus-Event auslösen
	nur nach dem kompletten Laden des Dokumentes
	vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getBoundingClientRect()	liefert Kooerinatens des Rechteckes um den Link
	Funktionswert ist Zeiger auf die Rechteck-Instanz vom Objekt-Typ TextRectangle
	mit den Eigenschaften
	.left
	.top
	.right
	.bottom
	alles Pixelpositionen
	nur IE ab 5.x
.handleEvent()	siehe Eventbehandlung zum NS ab 4.x
	nur NS ab 4.x
.releaseCapture()	siehe Eventbehandlung zum IE ab 5.x
	nur IE ab 5.x
.reload([true])	wenn true entfällt: Dokument vom Cache auf Festplatte laden
	wenn true kodiert: Dokument vom Server und nicht Cache laden
	Achtung: Server kann selbst Cache haben und
	daraus das Dokument liefern



.replace(url)	aktuelle Seite mit neuer geladener Seite überschreiben sowie den History-Eintrag zur aktuellen Seite überschreiben (keinen neuen bilden) --> BACK-Button wechselt danach nicht zur überschriebenen Seite, da diese in der History nicht mehr bekannt ist
.scrollIntoView(true oder false)	Oberkante des Links in den sichtbaren Bereich scrollen true, so bis zum oberen Fensterrand false, so bis zum unteren Fensterrand
.setCapture()	nur IE ab 5.x siehe Eventbehandlung zum IE ab 5.x nur IE ab 5.x

link Objekt des Internet Explorer:

Das Objekt kann nur innerhalb <HEAD> ... <HEAD> benutzt werden und wird nicht angezeigt. Es dient zum Einbinden externer Dateien, die den Quellcode des Dokumentes manipulieren sollen.

Beispiel:

```
<LINK REL=stylesheet HREF="styles.css" type="text/css">
```

Eigenschaften:

.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.charset	Zeichensatz zum Encoden eines Objektes im Dokument
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.href	Ziel-Url oder Anker als Sprungziel (z.B. <A>-Tag) siehe auch Eigenschaften .rel und .rev
.hreflang	Sprachcode des Objektes laut RFC1766
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.rel	Beziehung zwischen Objekt in Quellseite und Nachfolger-Zielseite laut Objekt Objekt ist ein Link z.B. A-Tag oder Link-Tag nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen beim durchforsten der Seiten und Nachfolgerseiten vorrangig kodiert in <A> oder <LINK> es muss zugleich die Eigenschaft .href kodiert und mit gültigen Wert belegt sein nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)
.rev	Beziehung zwischen Objekt in Quellseite und Vorgänger-Zielseite laut Objekt Objekt ist ein Link z.B. A-Tag oder Link-Tag nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen beim durchforsten der Seiten und Vorgängerseiten vorrangig kodiert in <A> oder <LINK> es muss zugleich die Eigenschaft .href kodiert und mit gültigen Wert belegt sein nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)
.scopeName	Namensraum laut XMLNS-Attribut
.sourceIndex	Index des Objektes in der Collection document.all
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.target	Name des Ziel-Fenster bzw. Ziel-Frame
.type	MIME-Typ des Objektes (Multipurpose Internet Mail Extension) wenn die Eigenschaft .classid nicht kodiert wird, dann immer .type kodieren bzw.



	die im Browserstandard enthaltenen MIME verwenden
	Hinweis: MIME wird von Simple Mail Transfer Protocol (SMTP) benutzt z.B. bei Audio, Images, Video, Texten
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbare
.cloneNode()	DOM wird geändert Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.detachEvent()	DOM nicht geändert Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.hasChildNodes()	DOM nicht geändert prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert



<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.removeAttribute()</code>	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.27. document.links Collection des Internet Explorer (HTML-Element mit HREF-Attribut)

Feld der Zeiger aller Objekte mit HREF-Eigenschaft (Attribut) sowie aller AREA-Objekte im Dokument,
wobei **alle** diese Objekte das Attribut NAME und oder ID besitzen **müssen**, um in dieser Collection referenziert zu sein.
Elementefolge laut HTML-Coding

Syntax:

```
[ var ZeigerAufFeld = ] document.links
[ var ZeigerAufFeldElement = ] document.links[Index]
```

Index Integer ab 0
muss in [] kodiert sein

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

<code>.item()</code>	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <code><INPUT TYPE=image ...></code> da dafür die children-Collection verwendet werden muss !!!
<code>.namedItem()</code>	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
<code>.tags()</code>	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM
<code>.urns()</code>	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.28. map Objekt des Internet Explorer

Das map-Objekt wird anhand eines img Objektes auf dem Client gebildet und stellt eine Sammlung (Mappe) von logischer Unterteilungen des Bildes in Teilbereiche dar. Jeder Teilbereich kann vom User agiert werden. Z.B. Selektive Auswahl des Users anhand der Teilbereiche des Bildes, um Bildteile in einem Extrafenster darstellen zu können. Teilbereiche werden nicht gerendert. Man sieht also nur das Bild als Gesamtheit.

Zur Bildung der Mappe wird der Anker als Wert des Attributes USEMAP benutzt.

Beispiel: ``

DerAnker stellt den freiwählbaren Mappennamen dar. Zum schnelleren Rendern des Bildes sollten die Attribute WIDTH und HEIGHT kodiert werden.

Die Mappe selbst wird durch Bezug auf den Anker kodiert.

Beispiel: `<MAP NAME="freier_mappen_name">`



Zur Bildung eines Teilbereiches muss der Koordinatenbereich innerhalb der Bilddimensionen festgelegt werden. Dazu dient das area Objekt. Es sind beliebig viele Teilbereiche festlegbar, die alle die Dimension des Bildes einhalten müssen.

Beispiel: <AREA SHAPE="rect" COORDS="0,0,82,126" ALT="Teilbereich 1" HREF="/graphics/test.gif">

Die Aktionsmöglichkeiten des Users können z.B. per HREF-Attribut und optional per Eventhandler kodiert werden. Da Teilbereiche nicht gerendert werden, sollt als Hilfe für den User das ALT-Attribut kodiert werden, um einen Text zu sehen, der erscheint, wenn die Maus über den Teilbereich fährt.

Beispiel:

```
<IMG SRC="test.gif" WIDTH=504 HEIGHT=126 BORDER=0 ALT="Gesamtes Bild" USEMAP="# freier_mappen_name">
<MAP NAME="freier_mappen_name">
  <AREA  SHAPE="rect"
    COORDS="0,0,82,126"
    ALT="Teilbereich 1"
    HREF="/graphics/test.gif"
    STYLE="....."
    onxxx=" ....."
  >
  ....
  <AREA ....."
</MAP>
```

Eigenschaften:

.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
.innerHTML	Zeiger aus ID bilden var Zeiger = eval(object.id); Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc. Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem



	des Elternobjektes (.offsetParent)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern



	DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
	DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByTagName()
	DOM nicht geändert
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
	DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
	DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
	DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
	DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
	DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
	DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
	DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
	DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
	DOM wird geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert



<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
<code>.swapNode()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.29. map.areas Collection des Internet Explorer

Zeigerfeld aller area Objekte eines map Objektes (siehe auch dort)

Erzeugung eines Elementes kann nur erfolgen durch die Methoden

`.createElement()` mit anschliessendem `.add()`

oder `.insertAdjacentHTML()`

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_map_objekt.areas
[ var ZeigerAufFeldElement = ] zeiger_auf_map_objekt.areas[Index , SubIndex]
```

<code>zeiger_auf_map_objekt</code>	laut ID-Attribut
<code>Index</code>	Integer ab 0 oder String z.B. laut ID-Attribut muss in [] kodiert sein
<code>SubIndex</code>	optional Unterindex also Unterelement eines Elementes nur kodieren wenn Index ein String ist Integer
<code>ZeigerAufFeld</code>	ist null, wenn keine area Objekte vorhanden
<code>ZeigerAufFeldElement</code>	ist null, wenn Feldelement nicht vorhanden

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

<code>.add()</code>	ein bereits erzeugtes Element einer Collection hinzufügen hinzufügen erst nach dem kompletten Laden des Dokumentes Element erzeugen per Methode <code>.createElement()</code>
<code>.item()</code>	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <code><INPUT TYPE=image ...></code> da dafür die <code>children</code> -Collection verwendet werden muss !!!
<code>.namedItem()</code>	Referenz auf Eintrag (Feldelement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
<code>.remove()</code>	ein Element entfernen aus einer Collection
<code>.tags()</code>	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM
<code>.urns()</code>	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.30. marquee Objekt des Internet Explorer

Scrolltext, der horizontal oder vertikal in den Dimensionen des Elternobjektes laufen kann.

Standardbreite ist die des Elternobjektes. Breite sollte kodiert werden, wenn marquee Objekt innerhalb TD liegt und der TD-Tag keine Breitenangabe enthält.

Für vertikales Scrollen ist die Eigenschaft `.scrollLeft` auf 0 zu setzen.

Für horizontales Scrollen ist die Eigenschaft `.scrollTop` auf 0 zu setzen.

<code>.behavior</code>	'scroll' Default. Scrollrichtung laut <code>.direction</code>
<code>'alternate'</code>	
<code>'slide'</code>	Scrollrichtung laut <code>.direction</code> , kein reinscrollen



Die Eigenschaften .scrollLeft und .scrollTop sind nur beschreibbar, wenn das Objekt **nicht** scrollt (Methode .start() noch nicht aktiviert bzw. Methode .stop() wurde aktiviert).

Die Methode .start() löst das Event onstart leider **nicht** aus.
Die Methode .stop() löst das Event onstop leider **nicht** aus.

Beispiel 1:

```
<MARQUEE          DIRECTION=RIGHT
                  BEHAVIOR=SCROLL
                  SCROLLAMOUNT=10
                  SCROLLDELAY=200
>
    Dieser Text scrollt
</MARQUEE>
```

Beispiel 2:

```
<MARQUEE          ID="ID_marquee"
                  DIRECTION=right
                  WIDTH=200
                  HEIGHT=200
                  STYLE="border-width:2px;border-style:solid;"
>
    Dieser Text scrollt rechts
</MARQUEE>
<BR>
<BUTTON onclick="alert(      'scrollLeft: ' + ID_marquee.scrollLeft
                              + ' scrollRight: ' + ID_marquee.scrollRight
                              + ' scrollTop: ' + ID_marquee.scrollTop
                              )
    "
>
    Anzeigen
</BUTTON>
<BUTTON onclick="ID_marquee.stop();ID_marquee.scrollLeft = 190;">
    Stop und scrollLeft=190
</BUTTON>
<BUTTON onclick=" ID_marquee.start();">
    Start
</BUTTON>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.behavior	Verhalten des Scrollens des marquee Objektes
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen Hintergrundfarbe des Objektes bzw. Dokumentes
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.direction	Start-Scrollrichtung des marquee Objektes
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden



	ab IE 6.x
	alternativ: Eigenschaft .dur
	siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
	siehe Objekt currTimeState und Behavior .style.time2
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
	Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.loop	Anzahl der Scrollaktionen des marquee Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern
	für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x
	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag
	wirksam mit parsen des Ende-Tag
	nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt
	nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollAmount	Scrollschrittweite in Pixel des marquee Objektes
.scrollDelay	Wartezeit in Millisekunden und laut Eigenschaft .trueSpeed zwischen zwei Scrollschritten
	eines marquee Objektes
	Scrollgeschwindigkeit, Fließgeschwindigkeit
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren
	sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes



	Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollTop	Abstand von oberem sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.trueSpeed	zeitliche Takteinheit zur Erzeugung der Pause laut Eigenschaft .scrollDelay eines marquee Objektes
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein



.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName() DOM nicht geändert
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann



	nur nach dem kompletten Laden des Dokumentes
	DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.start()	Start der Scrollaktion eines marquee Objektes Anzahl der Scrollaktionen laut Eigenschaft .loop erzeugt nicht das Ereignis onstart



Hinweis: Eigenschaften .scrollLeft und .scrollTop sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.

.stop() Stopp der aktuellen und laut Eigenschaft .loop noch offener Scrollaktionen eines marquee Objektes erzeugt **nicht** das Ereignis onstop

Hinweis: Eigenschaften .scrollLeft und .scrollTop sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.

.swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
nur sichtbar wenn Endetag geparkt
DOM wird geändert

4.3.2.2.4.3.31. meta Objekt des Internet Explorer

dient der Beschaffung von Informationen über Dokument, Client und Server durch Suchmaschinen, Browser und Server
nur im HEAD des Dokumentes kodierbar
nicht verschachtelbar

Kodierung des META-Objektes in HTML

```
<META HTTP-EQUIV="....." CONTENT="....." SCHEME=".....">
oder <META NAME="....." CONTENT="....." SCHEME=".....">
```

HTTP-EQUIV und NAME beschreiben die Art der Information
CONTENT enthält den Wert der Information
SCHEME-Attribut ist optional

Beispiel: Dokument für Suchmaschine vorbereiten

```
<META NAME="description"
oder NAME="author"
oder NAME="keywords"
oder NAME="date"
oder NAME="robots"

CONTENT="text_bzw_angaben_zu_name"
LANG="xx"

>
```

Beispiele:

```
<META NAME="author" CONTENT="Ich" >
<META NAME="date" CONTENT="Mittwoch, 26. April 2000">
<META NAME="description" CONTENT="freier text der von suchmaschinen indiziert wird">
<META NAME="keywords" LANG="xx" CONTENT="stichwortliste freier art">
```

Hinweis zum Attribut LANG:

Verwendung für mehrsprachige Stichwortliste
lang="xx" z.B. "de" für Deutschland
"en" für Englisch

Bsp.:

```
<META NAME="keywords" CONTENT="....." LANG="de">
<META NAME="keywords" CONTENT="....." LANG="en">

<META NAME="revisit-after" CONTENT="20 days">
```

Suchmaschine beim Scannen des Dokumentes beeinflussen:

Suchmaschine darf alles tun

```
<META HTTP-EQUIV="Robots" CONTENT="all" >
```

Suchmaschine darf Dokument nichts scannen

```
<META HTTP-EQUIV="Robots" CONTENT="noindex" >
```

Suchmaschine darf Dokument scannen

```
<META HTTP-EQUIV="Robots" CONTENT="index" >
```

Suchmaschine darf Verweisen folgen

```
<META HTTP-EQUIV="Robots" CONTENT="follow" >
```

Suchmaschine darf Verweisen nicht folgen

```
<META HTTP-EQUIV="Robots" CONTENT="nofollow" >
```

Beispiel: Um sicher zu gehen, daß immer die Seite mit garantiert aktuellem Stand geladen wird

--> eventuell ist Proxyserver nicht aktuell

```
<META HTTP-EQUIV="expires" CONTENT=sekunden_wert>
```

sekunden_wert: Wartezeit in Sekunden bis zum nächsten Laden unter Umgehung des Proxyserver-Cache --> 0, also immer umgehen und sofort laden

Tag und Zeitpunkt an dem unter Umgehung des Proxyserver-Cache geladen wird



"day, tt mon jjjj hh:mm:ss GMT"

Bsp: "Sat, 14 Dec 1999 12:00:00 GMT"

day: Mon Tue Wed Thu Fri Sat Sun

mon: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

TITLE-Tag kodieren

falls dieser nicht kodiert, so "No Title" von der Suchmaschine verwendet

jede Seite MUSS einen Rücksprung zur Startseite enthalten, da eine Suchmaschine Links auf der Seite abklappert

Beispiel für die Nutzung von browsereigenen Erweiterungen ab IE 4.x:

HTML-Seite ein- bzw. ausblenden per Filter

```
<META HTTP-EQUIV="Page-Enter" content="revealTrans(Duration=2.0, Transition=x)">
<META HTTP-EQUIV="Page-Exit" content="revealTrans(Duration=2.0, Transition=x)">
```

mit x	für	0	sich schliessender Kasten
		1	sich öffnender Kasten
		2	sich schliessender Kreis
		3	sich öffnender Kreis
		4	Vorhang nach oben
		5	Vorhang nach unten
		6	Vorhang nach rechts
		7	Vorhang nach links
		8	vertikaler Streifen
		9	horizontaler Streifen
		10	Schachbrettmuster nach rechts
		11	Schachbrettmuster nach unten
		12	zerbröseln
		13	Vorhang senkrecht zu Mitte hin
		14	Vorhang senkrecht von Mitte weg
		15	Vorhang horizontal zu Mitte hin
		23	Zufallsform

Beispiel für Vermeidung von Smart-Tags ab IE 6.x:

```
<META NAME="MSSmartTagsPreventParsing" CONTENT="TRUE">
```

Beispiel Dokument alle 2 Sekunden neu laden

```
<META HTTP-EQUIV="REFRESH" CONTENT=2>
```

Beispiel Zeichensatz des Dokumentes

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=Windows-1251">
```

Beispiel Themenunterstützung abschalten

```
<META HTTP-EQUIV="MSTHEMECOMPATIBLE" CONTENT="no">
```

Beispiel Grafik-Smarttag abschalten ab IE 6.x (Deaktivierung der automatischen Symbole bei Maus-Over über ein Bild)

```
<META HTTP-EQUIV="IMAGETOOLBAR" CONTENT="no">
```

Alternative: pro IMG das Attribut GALLERYIMG mit Wert "yes" bzw. "no" kodieren

Eigenschaften:

.charset	Zeichensatz zum Encoden eines Objektes im Dokument
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.content	Wert der Informationen laut Eigenschaften .httpEquiv und .name des meta Objektes
.disabled	Interaktionsfähigkeit



.httpEquiv	nur wenn sichtbar so User-Interaktion möglich Informationen des HTTP Response Header per meta Objekt Beschaffung der Informationen per Serverfunktionen HttpQueryInfo und QueryInfo Dieser Kopf wird von Suchmaschine, Server und Browser verwendet der Wert der Information wird in der Eigenschaft .content abgelegt
.isTextEdit	Erzeugbarkeit eines Textbereiches
.name	Informationen diverser Arten per meta Objekt der Wert der Information wird in der Eigenschaft .content abgelegt
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.scheme	Schema der Interpretation des Wertes laut Eigenschaft .content per meta Objekt z.B. von Datum und Zeit oder Tag-Content-Beschreibung
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrolling	Scrollenbar erzeugen
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.tagName	Tag-Bezeichner des Objektes
Methoden:	
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.remove()	ein Element aus einer Collection entfernen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert

4.3.2.2.4.3.32. noFrames Objekt des Internet Explorer

Container für denjenigen Code der Homepage, der nur durch Browser abgearbeitet wird, die FRAMESET-Tag nicht kennen.

Beispiel:

```
<FRAMESET>
  <NOFRAMES>
    Ihr Browser kennt das FRAMESET-Tag nicht
  </NOFRAMES>
</FRAMESET>
```

Zugriff auf noFrames-Eigenschaft:

Beispiel

```
var Wert = document.all.ZeigerAufnoFrames.eigenschaft;
```

mit ZeigerAufFrameset laut ID-Attribut

```
<NOFRAMES ID="ZeigerAufnoFrames" ..... >
```

Eigenschaften:

.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout



	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.fireEvent()	ein Event auslösen
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert

4.3.2.2.4.3.33. noScript Objekt des Internet Explorer

Container für denjenigen Code der Homepage, der nur durch Browser abgearbeitet wird, die das SCRIPT-Tag nicht kennen.

Zugriff auf noScript-Eigenschaft:

Beispiel

```
var Wert = document.all.ZeigerAufnoScript.eigenschaft;
```

mit ZeigerAufFrameset laut ID-Attribut

```
<NOSCRIPT ID="ZeigerAufnoScript" ..... >
```

Eigenschaften:

.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.fireEvent()	ein Event auslösen
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)



DOM wird geändert
 .setAttributeNode() Attribut einem Knoten zuweisen und Referenz liefern
 DOM wird geändert

4.3.2.2.4.3.34. object Objekt des Internet Explorer

Container eines per OBJECT-Tag eingebetteten Objektes aus dem HEAD und /oder BODY-Teil des Dokumentes (z.B. ActiveX-Control), also nicht eines per new-Methode erzeugten Objektes.

Das Objekt ist **nicht** das JScript-Objekt object.

Dieser Container ist notwendig, um private Methoden und Eigenschaften des Objektes von den weiter unten genannten Methoden und Eigenschaften des OBJECT-Tags im DOM beim Zugriff unterscheiden zu können.

Bsp: Methode .click() ist eine im DOM implementierte Methode zum OBJECT-Tag
 Wenn dem Objekt eine privaten Methode .click() implementiert ist, so erfolgt der Zugriff auf diese private Methode wie folgt:

```
document.all.objectID.object.click()
```

oder var ZeigerAufObjekt = document.all.objectID;
 ZeigerAufObjekt.object.click();

wobei objectID z.B. aus dem ID-Attribut des OBJECT-Tags stammt.

Dieser Container verhindert das Überschreiben von im DOM gleichnamigen Methoden und Eigenschaften.

Er sollte immer verwendet werden, vorallem dann, wenn der DOM des Browsers sich geändert hat und der Programmierer davon nichts weiss.

Wenn Objekt nicht instanziiert werden kann, so werden trotzdem alle innerhalb <OBJECT> </OBJECT> renderbaren Elemente angezeigt.

Events werden immer direkt zum Objekt gesendet !
 Events sind per Script programmierbar.

siehe auch document.embeds Collection

Hinweis zu Direct Animation per ActiveX-Control:

Für das STYLE-Attribut des OBJECT-Tags ist **dringend anzuraten**, es **nicht** zu kodieren, sondern **alle** Style-Angaben ausschliesslich durch Referenz per zeiger_auf_da_objekt.style.style_wert =; zu erzeugen. Grund: Je nach DA-Objekt-Implementierung wird das STYLE-Attribut im OBJECT-Tag teilweise oder vollständig ignoriert und damit wirkungslos. Der

Zeiger auf das DA-Objekt ist der Wert des ID-Attributes im OBJECT-Tag.

Beispiel:

```
<OBJECT CLASSID=" ..... " .....>
  <SPAN STYLE="color:red">
    Das Objekt ist nicht vom Browser instanzierbar !
  </SPAN>
</OBJECT>
```

Beispiel 1

```
<SCRIPT FOR= objectID EVENT= eEvent>
  :
</SCRIPT>
```

Beispiel 2

```
<OBJECT ID="objectID" CLASSID="xyz.abc">
</OBJECT>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
.altHTML	HTML-Script, der ausgeführt wird, wenn Objekt nicht geladen werden kann
.archive	Zeichenkette für Archive-Funktionalität eines Objektes
.BaseHref	Url des Dokumentes, das <OBJECT> </OBJECT> enthält (auch frame und iframe) ist meistens die Url des Dokumentes selbst
.border	Rahmendicke in Pixel
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.classid	Klassenbezeichner (ID) eines damit dem Objekt zugeordneten ActiveX-Controls von Microsoft Windows (Microsoft eigene oder Fremdhersteller)



	dient als Ersatz für die browserspezifischen MIME-Typen
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.code	Url der *.class-Datei (kompilierter Javacode)
.codeBase	Url der Komponente für Download der Komponente vom Server auf den Client optionale Versionsprüfung möglich (nicht bei Applet-Komponente) um unnötigen Download zu ersparen: vorheriger Abgleich der Version der Komponente auf Server und Client auch wenn der Versionsabgleich keinen Download ergab, wird immer HTTP Header-Transaktion ausgelöst Internet Media Type (Mimetype) des Codes zum Objekt, das per OBJECT-Tag eingebunden
.codeType	
wurde	
.data	Url der Daten des Objektes
.declare	Zeichenkette für Declare-Funktionalität des Objektes, das per OBJECT-Tag eingebunden wurde
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.object	Zeiger auf das Objekt laut Applet bzw. laut Objekt object
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordset	Zeiger des Standard-Record-Set in einem Datasource-Objekt (DSO) Methode .namedRecordset() liefert den Zeiger für Datenquellen-Objekt mit Namen



	also eines beliebige Mitgliedes im Datasource-Objekt (DSO)
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.standby	Zeichenkette für Standby-Funktionalität des per OBJECT-Tag eingebundenen Objektes
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.type	MIME-Typ des Objektes (Multipurpose Internet Mail Extension) wenn die Eigenschaft .classid nicht kodiert wird, dann immer .type kodieren bzw. die im Browserstandard enthaltenen MIME verwenden Hinweis: MIME wird von Simple Mail Transfer Protocol (SMTP) benutzt z.B. bei Audio, Images, Video, Texten
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.useMap	Url oder Anker für client-seitige Image-Map
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus



<code>.cloneNode()</code>	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.detachEvent()</code>	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_</code> bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
<code>.dragDrop()</code>	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
<code>.fireEvent()</code>	ein Event auslösen
<code>.focus()</code>	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
<code>.getAdjacentText()</code>	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
<code>.getAttribute()</code>	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
<code>.getAttributeNode()</code>	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
<code>.getBoundingClientRect()</code>	Referenz auf TextRectangle-Objekt im Element holen
<code>.getClientRects()</code>	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
<code>.getExpression()</code>	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.namedRecordset()</code>	Zeiger desjenigen Datenquellen-Record-Objektes mit Namen, das den Standard-Record-Set enthält Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an Eigenschaft <code>.recordset</code> liefert den Zeiger für den Standard-Record-Set in einem Datasource-Objekt (DSO)
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind: onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode <code>.setCapture()</code>
<code>.removeAttribute()</code>	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)



.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceNode()	DOM wird nicht geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird
.setActive()	Objekt muss an sich schon renderbar sein Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setCapture()	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
.setExpression()	ab IE 5.5 Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.35. document.embeds Collection des Internet Explorer (vermutlich auch im IE 6.x)

Feld der Zeiger aller als Plugin per EMBED-Tag eingebetteten Objekte im Dokument (inkl. Applets)
Elementefolge laut HTML-Coding

Achtung: Ab IE 6.x werden keine Plugins mehr unterstützt, damit ist die plugins Collection hinfällig. Inwieweit diese Collectionen noch implementiert ist oder bleibt, ist durch den Programmierer zu prüfen. Ab dem IE 6.x muss jedes Plugin, das in das Dokument eingebunden werden soll, durch ein ActiveX-Control implementiert werden. Plugins, wie sie beim Netscape existieren, laufen unter dem IE ab 6.x nicht mehr.

Es ist zu vermuten, dass im IE ab 6.x diese Collection nur eine Dummy-Funktion hat, also existiert, aber nie angefasst wird.

Diese Collection ist ein Alias für die plugins Collection **nur** bezüglich von Plugins (und nicht anderen einbettbaren Objekten), wobei letztere nur der Kompatibilität mit anderen plugin-fähigen Browsern dient. Alle eingebetteten Objekte haben in `document.embeds` ihren Zeiger (inklusive Plugins, Ausnahme: siehe tiefer).

Das Ansprechen von Plugins kann über die Collection `navigator.mimeTypes` per Eigenschaft `.enabledPlugin` erfolgen, die einen Zeiger auf das installierte Plugin enthält (wenn nicht installiert, so null-Zeiger). Diese Collection muss aber beim Internet Explorer nicht komplett gefüllt sein, kann aber (ausprobieren). Wenn der Zeiger nicht null ist, dann sind die plugin-eigenen Methoden und Eigenschaften über Punktnotation ansprechbar.

Es ist empfehlenswert, beim Internet Explorer **nur** mit der `document.embeds` Collection zu arbeiten. Die Collection `navigator.mimeTypes` kann zwar unter NS und IE mit dem gleichen Script-Code angesprochen werden, aber das ist spätestens ab IE 6.0 nicht mehr möglich.

Erzeugung:
durch Browser

Beispiel für EMBED-Tag beim IE unter 6.x und beim NS:

```
<EMBED
  SRC="url"
  TYPE="mime_typ"           // z.B. "image/gif"
  PLUGINSOURCE="plugin_url"
  HEIGHT="hoehe_plugin_objekt"
  WIDTH="breite_plugin_objekt"
  NAME="ID_Embed"
  HIDDEN="true oder false"  // true so Objekt unsichtbar
>
<PARAM Name="parameter_name" VALUE=parameter_wert>
```



```
.....
</EMBED>
```

Syntax:

```
[ var ZeigerAufFeld = ] document.embeds
[ var ZeigerAufFeldElement = ] document.embeds[Index, [SubIndex]]
```

Index	oder	Integer ab 0 String Name oder ID des Elementes muss in [] kodiert sein
SubIndex		optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

```
ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()
```

beim IE:

```
document.all.ID_Embed.eigenschaft
document.all.ID_Embed.methode()
```

beim NS:

```
document.ID_Embed.eigenschaft
document.ID_Embed.methode()
```

Eigenschaften beim Internet Explorer:

```
.length      Anzahl der Feldelemente also Feldlänge
```

Methoden beim Internet Explorer:

```
.item()      Referenz auf Feldelement anhand des Integer-Indexes oder des
              Attributnamen (analog zu ID oder NAME-Attribut) liefern
              außer bei Formular mit <INPUT TYPE=image ...>
              da dafür die children-Collection verwendet werden muss !!!
.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
              (analog zu ID oder NAME-Attribut) liefern
.tags()      Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
              siehe tags Collection des DOM
.urns()      Referenz auf Feld aller Elemente mit gemeinsamer URN liefern
```

4.3.2.2.4.3.36. document.select Objekt des Internet Explorer

Drop-Down-Liste oder Liste in einer Box (List-Box) als Objekt:

Die Zeiger aller Optionen werden in der Collection document.select.options gesammelt.

Ab IE 5.5 werden option Objekte zusätzlich auch über die Collection **document.all** referenzierbar.Achtung: .z-index Attribut wird **nicht** unterstützt.

Beispiel 1:

```
<SELECT NAME="Katzen" SIZE="1">
  <OPTION VALUE="1">Mischling
  <OPTION VALUE="2">Siamese
  <OPTION VALUE="3" SELECTED>Kratzkehl
</SELECT>
```

Beispiel 2:

```
<SELECT NAME="Autos" SIZE="3" MULTIPLE>
  <OPTION VALUE="1" SELECTED>BMW
  <OPTION VALUE="2">Porsche
  <OPTION VALUE="3" SELECTED>Mercedes
</SELECT>
```

Beispiel 3:

```
<SCRIPT LANGUAGE="JScript">
  var OptionObjekt = document.createElement("OPTION");
  OptionObjekt.text="Ferrari";
  OptionObjekt.value="4";

  // Annahme: selection Objekt sei vorhanden
  SelektionObjekt.add(OptionObjekt);
</SCRIPT>
```

Beispiel 4: Leere Textarea per Textwert einer selektierten Option aus Drop-Down-Liste füllen:

```
<SCRIPT LANGUAGE="JScript">
  function TextAreaErweitern()
  {
    // Index der selektierte Option holen
    var Index = ID_Select.selectedIndex;

    // Text-Wert der selektieren Option holen
```



```

        var TextWert = ID_Select.options[Index].text;

        // Textwert an Textarea anhängen
        ID_Textarea.value+= TextWert + "\n";
    }
</SCRIPT>
<SELECT ID="ID_Select" SIZE="1" onchange="TextAreaErweitern(">
    <OPTION VALUE="1">BMW
    <OPTION VALUE="2">PORSCHE
    <OPTION VALUE="3" SELECTED>MERCEDES
</SELECT>
<TEXTAREA ID="ID_Textarea"></TEXTAREA>

```

Beispiel 5:

```

<SCRIPT>
    function AusschnittSetzen()
    {
        ID_Image.style.clip= "rect(0,100, "
            + ID_select.options(ID_select.selectedIndex).value
            + ",100)";

        if (ID_Image.currentStyle.clipBottom == "60px")
        { alert("60 Pixel"); }
    }
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen(">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>

```

Beispiel 6:

```

<SELECT DATASRC="#Anker" DATAFLD="KartenTyp">
    <OPTION>Visa
    <OPTION>Mastercard
    <OPTION>American Express
    <OPTION>Diner's Club
    <OPTION>Sparkasse
</SELECT>

```

Beispiel 7:

```

<SELECT ID="ID_select" MULTIPLE>
    <OPTION>Auswahl 1</OPTION>
    <OPTION> Auswahl 2</OPTION>
    <OPTION> Auswahl 3</OPTION>
</SELECT>
<BUTTON onclick="ID_select.multiple=false">keine multiple Auswahl</BUTTON>
<BUTTON onclick="ID_select.multiple=true">multiple Auswahl</BUTTON>

```

Beispiel 8:

```

<SCRIPT>
    var ZeigerAufNeueOption = document.createElement("OPTION");
    ID_Select.options.add(ZeigerAufNeueOption);
    ZeigerAufNeueOption.innerText = "Option 2";
    ZeigerAufNeueOption.Value = "2";
</SCRIPT>
<SELECT ID="ID_Select" >
    <OPTION VALUE="1">Option 1</OPTION>
</SELECT>

```

Beispiel 9:

```

<SCRIPT>
    function Erzeugen()
    {
        ID_Span.innerHTML="";
        var FeldDerZeigerAufOption = ID_Select.options[ID_Select.selectedIndex];

        if(FeldDerZeigerAufOption.text.length>0)
        {
            var ZeigerAufElement= document.createElement(FeldDerZeigerAufOption.text);
            eval(
                " ZeigerAufElement."
                + FeldDerZeigerAufOption.value

```




```

        + ""
        + ID_Input.value
        + ""
    );

    if(FeldDerZeigerAufOption.text=="A")
    { ZeigerAufElement.href="javascript:alert('A link.');" };
}
ID_Span.appendChild(ZeigerAufElement);
}
</SCRIPT>
<SELECT ID="ID_Select" onchange="Erzeugen()">
    <OPTION VALUE="innerText">A
    <OPTION VALUE="value">&lt;INPUT TYPE="button"&gt;
</SELECT>
<INPUT TYPE="text" ID="ID_Input" VALUE="Beispiel Text">
<SPAN ID="ID_Span" ></SPAN>

```

Zugriff:

```

document.all.zeiger_auf_select_objekt.eigenschaft
document.all.zeiger_auf_select_objekt.methode()

```

zeiger_auf_select_objekt z.B. laut ID-Attribut

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.ATOMICSELECTION	Selektierbarkeit einstellen
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.length	Anzahl der Feldelemente also Feldlänge der Collection document.select.options (siehe Eigenschaft .options)
.multiple	Multiple Auswahl bei document.select



.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von offsetHeight, offsetLeft, offsetTop und offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.options	Zeiger auf die Collection document.select.options des document.select Objektes
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.selectedIndex	Index der Option in der Collection document.select.options des Objektes select jedes option Objekt eines select Objektes wird im Feld, das Teil des select Objektes ist, referenziert es kann immer nur eine Option innerhalb einer Optionsgruppe selektiert sein
.size	Anzahl der Zeilen in einer List-Box als select Objekt
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen Achtung: Bezüglich der Style-Eigenschaften .background-color und .color kann nur genau 1 Style deklariert werden für alle Objekte document.select.option und das document.select Objektes insgesamt. Es wird der Style des ersten in der Kodierungsfolge gefundenen Objektes document.select.option für alle anderen und das Objekt document.select selbst verwendet. Es wird angeraten, nur im document.select das STYLE-Attribut zu verwenden. Das Objekt document.select.option kann nur genau folgende Style-Eigenschaften erhalten: .background-color und .color
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup

Anspringen default per TAB-Taste
für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT,
SELECT, TEXTAREA



	Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.type	prüfen auf multiples select Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektierbarkeit z.B. mit Maus wird nicht an Kinder vererbt bei Wechsel: vorhergehende vorhandene Selektion wird nicht verändert
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
Methoden:	
.add()	ein bereits erzeugtes Element einer Collection hinzufügen hinzufügen erst nach dem kompletten Laden des Dokumentes Element erzeugen per Methode .createElement()
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht



	Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByTagName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.remove()	ein Element aus einer Collection entfernen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient.



	Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
	DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt
	ersetzende Objekt muss per Methode .createElement() erzeugt worden sein
	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
	sichtbar erst mit parsen des Endetags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren
	aber ohne es zu fokussieren
	und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen
	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt
	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
	Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient
	Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
	nur sichtbar wenn Endetag geparkt
	DOM wird geändert
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.36.1. document.select.option Objekt des Internet Explorer

Objekt einer Option aus einer Selektion per document.select Objekt

Option ist anwählbar

Die Zeiger aller Optionen werden in der Collection document.select.options gesammelt.

Ab IE 5.5 werden option Objekte zusätzlich auch über die Collection **document.all** referenzierbar.

Zum Hinzufügen eines option Objektes müssen das select und option Objekt in einem **gemeinsamen** Fenster liegen.

Beispiel 1:

```
<SELECT NAME="Katzen" SIZE="1">
  <OPTION VALUE="1">Mischling
  <OPTION VALUE="2">Siamese
  <OPTION VALUE="3" SELECTED>Kratzkehl
</SELECT>
```

Beispiel 2:

```
<SELECT NAME="Autos" SIZE="3" MULTIPLE>
  <OPTION VALUE="1" SELECTED>BMW
  <OPTION VALUE="2">Porsche
  <OPTION VALUE="3" SELECTED>Mercedes
</SELECT>
```

Beispiel 3:

```
<SCRIPT LANGUAGE="JScript">
  var OptionObjekt = document.createElement("OPTION");
  OptionObjekt.text="Ferrari";
  OptionObjekt.value="4";

  // Annahme: selection Objekt sei vorhanden
  SelektionObjekt.add(OptionObjekt);
</SCRIPT>
```

Beispiel 4: Leere Textarea per Textwert einer selektierten Option aus Drop-Down-Liste füllen:

```
<SCRIPT LANGUAGE="JScript">
```



```

function TextAreaErweitern()
{
    // Index der selektierte Option holen
    var Index = ID_Select.selectedIndex;

    // Text-Wert der selektieren Option holen
    var TextWert = ID_Select.options[Index].text;

    // Textwert an Textarea anhängen
    ID_Textarea.value+= TextWert + "\n";
}
</SCRIPT>
<SELECT ID="ID_Select" SIZE="1" onchange="TextAreaErweitern()">
    <OPTION VALUE="1">BMW
    <OPTION VALUE="2">PORSCHE
    <OPTION VALUE="3" SELECTED>MERCEDES
</SELECT>
<TEXTAREA ID="ID_Textarea"></TEXTAREA>

```

Beispiel 5:

```

<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect(0,100, "
        + ID_select.options(ID_select.selectedIndex).value
        + ",100)";

    if (ID_Image.currentStyle.clipBottom == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>

```

Beispiel 6:

```

<SELECT DATASRC="#Anker" DATAFLD="KartenTyp">
    <OPTION>Visa
    <OPTION>Mastercard
    <OPTION>American Express
    <OPTION>Diner's Club
    <OPTION>Sparkasse
</SELECT>

```

Beispiel 7:

```

<SELECT ID="ID_select" MULTIPLE>
    <OPTION>Auswahl 1</OPTION>
    <OPTION> Auswahl 2</OPTION>
    <OPTION> Auswahl 3</OPTION>
</SELECT>
<BUTTON onclick="ID_select.multiple=false">keine multiple Auswahl</BUTTON>
<BUTTON onclick="ID_select.multiple=true">multiple Auswahl</BUTTON>

```

Beispiel 8:

```

<SCRIPT>
var ZeigerAufNeueOption = document.createElement("OPTION");
ID_Select.options.add(ZeigerAufNeueOption);
ZeigerAufNeueOption.innerText = "Option 2";
ZeigerAufNeueOption.Value = "2";
</SCRIPT>
<SELECT ID="ID_Select" >
    <OPTION VALUE="1">Option 1</OPTION>
</SELECT>

```

Beispiel 9:

```

<SCRIPT>
function Erzeugen()
{
    ID_Span.innerHTML="";
    var FeldDerZeigerAufOption = ID_Select.options[ID_Select.selectedIndex];

```



```

        if(FeldDerZeigerAufOption.text.length>0)
        {
            var ZeigerAufElement= document.createElement(FeldDerZeigerAufOption.text);
            eval(
                " ZeigerAufElement."
                + FeldDerZeigerAufOption.value
                + "="
                + ID_Input.value
                + ""
            );

            if(FeldDerZeigerAufOption.text=="A")
            { ZeigerAufElement.href="javascript:alert('A link.');" }

        }
        ID_Span.appendChild(ZeigerAufElement);
    }
</SCRIPT>
<SELECT ID="ID_Select" onchange="Erzeugen()">
    <OPTION VALUE="innerText">A
    <OPTION VALUE="value">&lt;INPUT TYPE="button"&gt;
</SELECT>
<INPUT TYPE="text" ID="ID_Input" VALUE="Beispiel Text">
<SPAN ID="ID_Span" ></SPAN>

```

Zugriff:

```

var Zeiger = document.all.zeiger_auf_select_objekt
Zeiger.zeiger_auf_option_objekt.eigenschaft
Zeiger.zeiger_auf_option_objekt.methode()

```

```

zeiger_auf_select_objekt z.B. laut ID-Attribut
zeiger_auf_option_objekt z.B. laut ID-Attribut

```

```

var ZeigerAufFeldElement = document.zeiger_auf_select_objekt.options [Index [, SubIndex]]

```

Index	Integer ab 0 oder String Name oder ID des Elementes (analog zu NAME und ID-Attribut) muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes

```

zeiger_auf_select_objekt            z.B. laut ID-Attribut

```

Eigenschaften:

.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.defaultSelected	Default-Selektionsstatus der Option, also Objektes document.select.option des Objektes document.select standardgemäß ist immer die oberste Option selektiert es kann aus einer Optionengruppe immer nur genau 1 Option selektiert sein
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.index	laufende Nummer der Option, also Objektes document.select.option, in einer List-Box als



	document.select Objekt
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.label	Label einer Option, also Objektes document.select.option des Objektes document.select
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von offsetHeight, offsetLeft, offsetTop und offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.selected	Selektionsstatus der Option, also Objektes document.select.option des Objektes document.select standardgemäß ist immer die oberste Option selektiert es kann aus einer Optionengruppe immer nur genau 1 Option selektiert sein direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen Achtung: Bezüglich der Style-Eigenschaften .background-color und .color kann nur genau 1 Style deklariert werden für alle Objekte document.select.option und das document.select Objektes insgesamt. Es wird der Style des ersten in der Kodierungsfolge gefundenen Objektes document.select.option für alle anderen und das Objekt document.select selbst verwendet. Es wird angeraten, nur im document.select das STYLE-Attribut zu verwenden. Das Objekt document.select.option kann nur genau folgende Style-Eigenschaften erhalten: .background-color und .color
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit).



	nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior)
	ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !
	siehe Objekt currTimeState und Behavior .style.time2
	siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemLanguage	Standard-Sprache des Betriebssystems (Sprache der Installation) per navigator Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.text	interner Kettenwert einer Option, , also Objektes document.select.option des Objektes document.select
	ändert nicht den angezeigten Text hinter <OPTION ...>
	ist nur ein interner Wert und beim Formular der gesendete Wert der Option
.timeContainer	Typ der Timeline des Objektes
	siehe Objekt currTimeState und Behavior .style.time2
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.value	Wert eines Objekt-Attributes
	Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
	im Formular: wenn Eigenschaft .text kodiert, so wird deren Wert gesendet und der Wert von
	.value bleibt nur angezeigt
	wenn Eigenschaft .text nicht kodiert, so wird der Wert von .value gesendet
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst
	werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
	DOM wird geändert
	Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
	Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag
	(falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz
	laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im
	im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
	NICHT verkettet sondern in Zufallsfolge , es sei denn
	die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen
	außer ID, STYLE und per Script definierte Attribute
	Script-erzeugte Attribute nicht entfernbar
	DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus
	manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern
	DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-
	Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
	auch für CSS-Layout
	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint()
	also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos
	erreicht haben
	Overbereich der Maus ist mehr als 1 Pixel gross
	beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern
	(Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
	DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler
	wobei Registrierung mit Methode .attachEvent() aktiviert wurde
	Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner
	zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt
	(also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
	Text kann HTML-Tags enthalten, muss aber nicht



.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.hasChildNodes()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
.insertAdjacentElement()	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.insertAdjacentHTML()	DOM wird geändert HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.insertBefore()	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu



	ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.setAttribute()	DOM wird geändert Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setCapture()	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.36.2. document.select.options Collection des Internet Explorer

Eine Option ist über dieses Feld der Zeiger aller Optionen anwählbar:

Die Zeiger aller Optionen werden in der Collection document.select.options gesammelt.

Ab IE 5.5 werden option Objekte zusätzlich auch über die Collection **document.all** referenzierbar.

Eine Option aus dem Feld löschen, erfolgt durch Zuweisung des Null-Zeiger (null). Zugleich wird automatisch das Feld kondensiert, also alle Lücken werden entfernt.

Zum Hinzufügen eines option Objektes müssen das select und option Objekt in einem **gemeinsamen** Fenster liegen.

Syntax:

```
[ var ZeigerAufFeld = ] document.zeiger_auf_select_objekt.options
[ var ZeigerAufFeldElement = ] document.zeiger_auf_select_objekt.options [Index [, SubIndex]]
```

Index	Integer ab 0
oder	String Name oder ID des Elementes (analog zu NAME und ID-Attribut)
	muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes
ZeigerAufFeldElement	ist null, wenn Feldelement nicht vorhanden wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection aus diesen Script-Objekten geliefert

zeiger_auf_select_objekt z.B. laut ID-Attribut

Beispiel:

```
var Feld = document.all.tags("SELECT");

if (Feld.length>0)
{
    for (var i=0; i < Feld[0].options.length; i++)
    {
        alert("Element " + i
            + " : mit internem Text = " + Feld[0].options(i).text
            + " und angezeigtem Wert = " + Feld[0].options(i).value);
    }
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen
hinzufügen erst nach dem kompletten Laden des Dokumentes
Element erzeugen per Methode .createElement()

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern



außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.remove() ein Element aus einer Collection entfernen

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.37. document.selection Objekt des Internet Explorer

Aktive Selektion eines Dokumentes markierter Textblock (document.selection.textrange Collection, Objekt textrange)
Control-Element(e) (document.selection.controlRange Collection) nur für body Objekt

Selektion erzeugbar durch User
Methode .select()

im Dokument kann zu jedem Zeitpunkt nur **genau 1** Selektion aktiv sein: ID-Attribut wird daher nicht unterstützt, weil nicht nötig

Leere Selektion entspricht: User setzt Cursor auf einen leeren Bereich im Dokument.

Elternobjekte mit Textrange sind body Objekt
button Objekt
textarea Objekt
input text Objekt
selection Objekt (nur wenn ein Text selektiert wurde (mit oder ohne HTML))

wobei diese weitere HTML-Elemente enthalten können, die ebenfalls Textbereiche besitzen können

Zugriff:

document.selection.eigenschaft
document.selection.methode()

Hinweis: Im Dokument kann zu jedem Zeitpunkt nur genau 1 Selektion aktiv sein. Deshalb reicht **document.selection** weil es eindeutig ist.

Eigenschaften:

.type Type der Selektion per document.selection Objekt des Dokumentes
siehe auch Methoden .createRange() .createControlRange() und .createTextRange()
Objekt textrange

.typeDetail Type der Selektion per document.selection Objekt des Dokumentes, wobei die Hostanwendung
diese Eigenschaft unterstützen und mit Wert belegen muss

Methoden:

.clear() Selektion als Markierung aufheben per document.selection Objekt des Dokumentes per document.selection
nicht Selektion löschen
siehe Methode .empty()

.createRange() Zeiger auf einen Universal-Bereich erzeugen per document.selection Objekt des Dokumentes
Universal-Bereich kann enthalten:
Text (document.selection.textrange Collection
textrange Objekt)
oder Control-Element(e) (document.selection.controlRange Collection)
siehe auch Methoden .createControlRange() .createTextRange() und .createRangeCollection()
Eigenschaft .type

.createRangeCollection() document.selection.textrange Collection erzeugen per document.selection Objekt des Dokumentes
nur wenn der Browser multiple Selektion unterstützt, so mehr als 1 Feld-Element textrange Objekt
vorhanden bei Mehrfach-Selektion durch User
siehe auch textrange Objekt
Methoden .createRange() .createControlRange() und .createTextRange()

.empty() Selektion löschen per document.selection Objekt des Dokumentes
setzt zugleich Eigenschaft
.item auf null von document.selection.textrange Collection
bzw. von document.selection.controlRange Collection
.type auf "none" von document.selection

4.3.2.2.4.3.37.1. document.selection.controlrange Collection des Internet Explorer

Feld der Zeiger aller Control-Elemente, die in der **aktiven** Selektion per document.selection Objekt markiert wurden

nur für body Objekt

ein Objekt controlrange existiert nicht

ab IE 5.x

Syntax:

```
[ var ZeigerAufFeld = ] document.selection.controlrange
[ var ZeigerAufFeldElement = ] document.selection.controlrange [Index]
```

Index Integer ab 0
oder String Name oder ID des Elementes
(analog zu NAME und ID-Attribut)
muss in [] kodiert sein

ZeigerAufFeldElement



ist null, wenn Feldelement nicht vorhanden
wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection
aus diesen Script-Objekten geliefert

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen
hinzufügen erst nach dem kompletten Laden des Dokumentes
Element erzeugen per Methode .createElement()

.execCommand() Kommando ausführen z.B. im aktuellen Dokument
in aktueller Selektion
im aktuellen Bereich
erst nach dem kompletten Laden des Dokumentes zulässig
Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
Control = Element zur Steuerung analog zum HTML-Element (Tag)
Input-Control = Element mit Eingabeeigenschaft

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.queryCommandEnabled() prüfen ob Kommando ausführbar ist
.queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
.queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird
.queryCommandValue() Wert eines Kommandos liefern
.remove() ein Element aus einer Collection entfernen
.scrollIntoView() Objekt derart scrollen, dass es im Fenster für User sichtbar wird
Objekt muss an sich schon renderbar sein

.select() Selektion eines Textranges (Textbereich, Objekt textrange) oder ControlRange (Control-Elemente)
nur unter Windows 32-Bit
siehe Objekt document.selection
Methoden .createTextRange() .createControlRange() und .createRange()

4.3.2.2.4.3.37.2. document.selection.textrange Collection des Internet Explorer

Feld der Zeiger aller textrange Objekte der **aktiven** Selektion

ab IE 5.5

Elternobjekte mit Textrange sind

body Objekt

button Objekt

textarea Objekt

input text Objekt

selection Objekt (nur wenn ein Text selektiert wurde (mit oder ohne HTML))

wobei diese weitere HTML-Elemente enthalten können, die ebenfalls Textbereiche besitzen können

Syntax:

[var ZeigerAufFeld =] document.selection.textrange

[var ZeigerAufFeldElement =] document.selection.textrange[Index [, SubIndex]]

Index Integer ab 0
oder String Name oder ID des Elementes
(analog zu NAME und ID-Attribut)
muss in [] kodiert sein

SubIndex optional
nur kodieren wenn Index ein String ist
Integer als Unterindex also Unterelement eines Elementes

ZeigerAufFeldElement
ist null, wenn Feldelement nicht vorhanden
wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection
aus diesen Script-Objekten geliefert

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.3.38. span Objekt

Das DIV- bzw. SPAN-Objekt ähneln sich sehr stark.. SPAN wird als üblicherweise als Textcontainer benutzt.

Beispiel: blauesWort



DIV bzw. SPAN können im Elternobjekt (Dokument im Fenster oder übergeordneter DIV bzw. SPAN) erzeugt werden,
als HTML-Element üblicherweise im BODY-Teil des Dokumentes
per Script durch z.B. document.write("<DIV >...");

Die Anzeige (das Rendern) des DIV bzw. SPAN erfolgt erst mit dem Parsen des Ende-Tags, also </DIV> bzw. . Danach können Komponenten zur Laufzeit nachträglich verändert werden, die dann aber nur z.T. sofort sichtbar werden (teilweise erst nach dem Reload des betreffenden Dokumentes).

NS und IE können das DIV- und SPAN-Objekt erzeugen, implementieren und rendern diese aber verschiedenartig.
Beispiel:

Nur der Internet Explorer erzeugt automatisch ein BODY-Objekt,
wenn dieses im Dokument nicht kodiert wurde, weil z.B.
sämtliche HTML-Elemente im HEAD des Dokumentes
per Script erzeugt wurden.

Der NS verlangt immer einen BODY-Teil und ignoriert o.g. Script-
Anweisungen im HEAD des Dokumentes, solange der
BODY des Dokumentes nicht komplett geparkt wurde
(</BODY> noch nicht erkannt wurde). Es können also
nur nachträglich neue HTML-Elemente per Script im
HEAD des Dokumentes erzeugt werden. Scriptaufrufe
aus dem BODY-Teil werden mit dem BODY geparkt, egal
ob die Scripte im HEAD oder BODY liegen.

Unter NS 6.x wird anstelle von DIV- bzw. SPAN-Objekt das LAYER-Objekt favorisiert. Ab NS 6.x wird das Layer-Objekt radikal nicht mehr unterstützt: Es sind für die Ebenendarstellung (überlappende Objekte) keine Layer mehr sondern z.B. DIV oder SPAN verwendbar.

Zugriff:

Die übliche Zugriffsweise ist über das ID-Attribut (Eigenschaft .id) des DIV bzw. SPAN.

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function HandlerFuerOnMoveStart()
{
    // anstelle des ID vom SPAN falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "SPAN wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandlerFuerOnMoveEnd()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

// 2-D Positionierung einschalten
document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
offsetLeft = <SPAN ID="ID_Span1"></SPAN>
<BR>
offsetTop = <SPAN ID="ID_Span2"></SPAN>
<BR>
<DIV CONTENTEDITABLE="true">
    <DIV STYLE=
        "position:absolute;width:300px;height:100px; background-color:red;"
    >
```



bewegbarer DIV

</DIV>

</DIV>

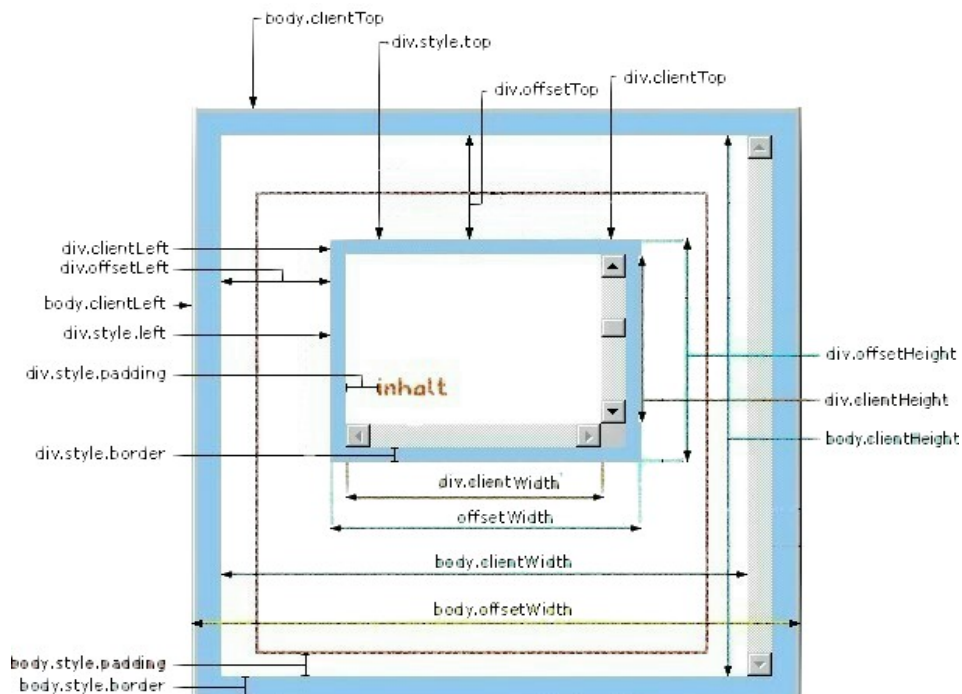
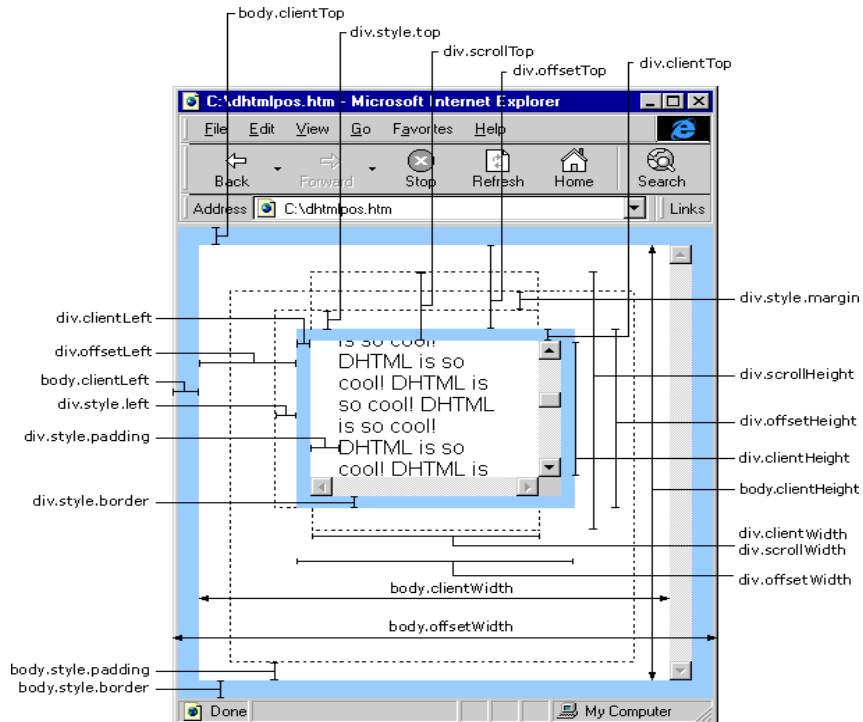
</BODY>

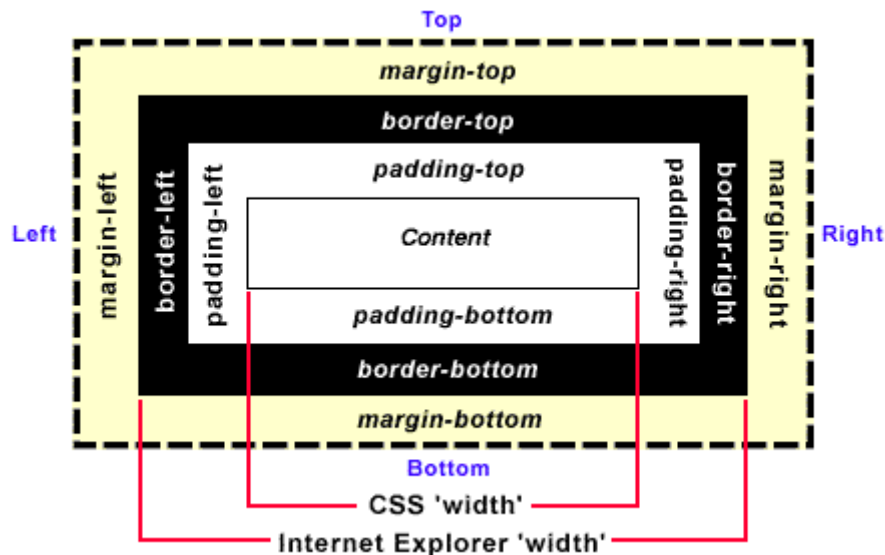
</HTML>

span Objekt des Internet Explorer:

Bezüglich zeitsteuernder Eigenschaften/Methoden werden auch Behavior style.time2 und Objekt currTimeState ab IE 6.x benutzt.

Wichtige Eigenschaften im IE:





Hinweis Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

Eigenschaften:

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem

Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !!!

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Defintion ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigeart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur



	siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
	siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2



.systemBitrate	siehe .syncTolerance und .syncBehavior
.systemCaptions	wird hier nicht erklärt
.systemLanguage	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	Standard-Sprache des Betriebssystems (Sprache der Installation) per navigator Objekt
.tabIndex	wird hier nicht erklärt
	Index des Elementes in der Tab-Tasten-Folge
	für Anspringen des Dokumentes
	Anspringen verbunden mit Focus erhalten
	--> Ereignisse werden ausgelöst !!
	unter IE 5.x
	onblur, onfocus
	ab IE 5.x
	onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste
	für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT,
	SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste
	für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes
	siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
	DOM wird geändert
	Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
	Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
	vor IE 5.0 TABINDEX-Attribut muss kodiert sein
	ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen
	außer ID, STYLE und per Script definierte Attribute
	Script-erzeugte Attribute nicht entfernen
	DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus
	manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern
	DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
	auch für CSS-Layout
	onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint()
	also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben
	Overbereich der Maus ist mehr als 1 Pixel gross
	beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
	DOM nicht geändert



.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Click auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein !
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt



	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.39. style Objekt des Internet Explorer

ab IE 4.x

z.T. erst ab IE 5.5 bzw. IE 6.x

Ansatz:

Aufgrund der HTML-Integration von CSS sollte der Browser CSS unterstützen. Ob diese Unterstützung gegenüber den aktuellen Standards zu HTML und CSS vollständig ist, hängt vom Willen des Browserherstellers ab. Letztendlich wird CSS sinnigerweise als grundlegende Komponente und Eigenschaft **aller** Objekte integriert. CSS bietet elementare Vielfalt bei der dynamischen Scriptprogrammierung, ist gewöhnungsbedürftig, aber durch Normung universell auf diverse Objekte anwendbar, die das STYLE-Attribut bzw. die Eigenschaft .style unterstützen.

Es sei darauf hingewiesen, dass

die Verwendung von CSS-Klassen, Kodierung von <STYLE> bzw. dem STYLE-Attribut bzw. der Eigenschaft .style nur **scheinbar**

alle synonym sind:

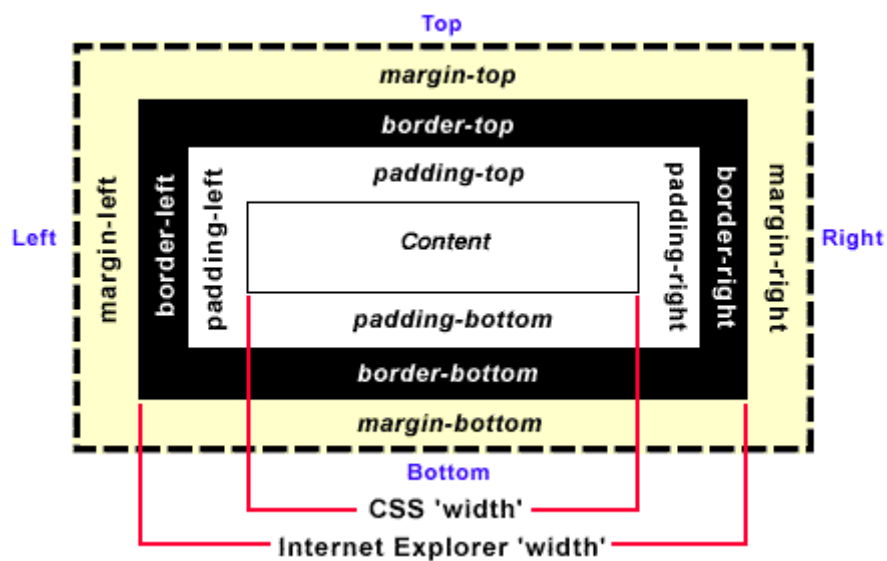
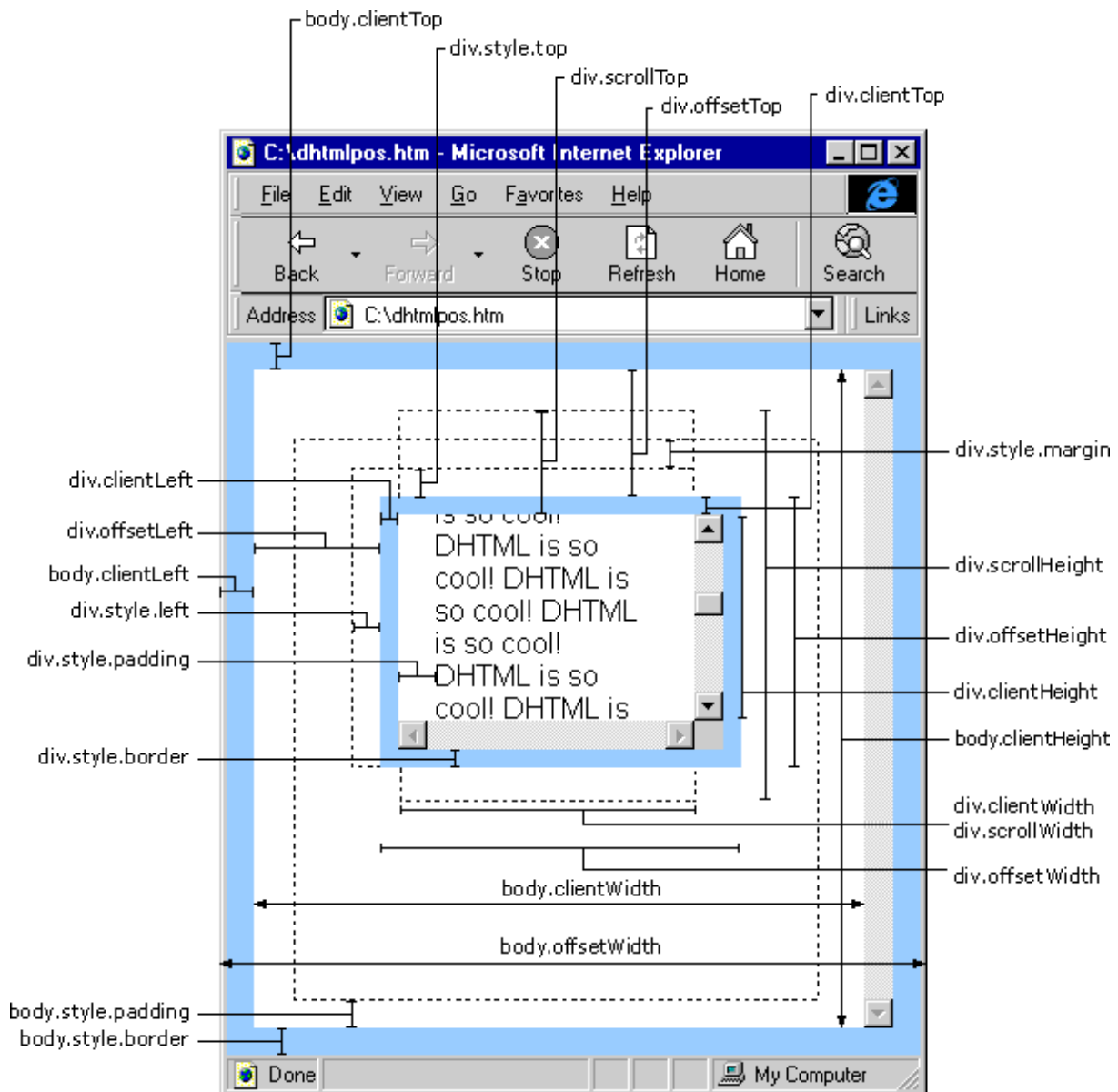
CSS-Klassen im HEAD können dokumentweit verwendet werden.

<STYLE>, STYLE-Attribut und .style sind stets HTML-komponentenbezogen.

der Browser diese Formen intern z.T. verschieden (auch bezüglich der Collectionen) verwaltet.



Das Einbinden von Styles aus einer externen Datei ist möglich.



style Objekt und styleSheet Objekt:

Beide Objekte behandeln zwar CSS, unterscheiden sich aber in der Aufgabe:

| | |
|-----------------------|--|
| Das Objekt style | repräsentiert CSS eines Objektes im HTML-Dokument
also z.B. das STYLE-Attribut im HTML-Tag
die Eigenschaft .style
alle Styles sind les- und schreibbar |
| Das Objekt styleSheet | repräsentiert CSS im HTML-DOM, also im gesamten Dokument
wird mit der document.styleSheets Collection verwaltet:
Feld der Zeiger aller styleSheet Objekte im Dokument
Syntax:
<pre>[var ZeigerAufFeld =] document.styleSheets [var ZeigerAufFeldElement =] document.styleSheets [index]</pre> <p>index Integer ab 0
muss in [] kodiert sein</p> <p>Eigenschaft .length Anzahl der Feldelemente
Integer, ab 1</p> |

Nachfolgende Beschreibungen orientieren sich praktischerweise hauptsächlich am Objekt style.

Beispiel: Es sollen DIVs erzeugt werden und deren Objektreferenzen (Zeiger) in einem Zeigerfeld gesammelt werden. Das Zeigerfeld dient der

vereinfachten Verwaltung der dynamisch erzeugten DIV's.

```
var DIV_ID="Test_DIV"; // auch "Otto_DIV" etc.möglich , je nach Wunsch

var DIV_Zeiger=Array(); // sammelt alle ID als Zeichenketten

// DIV's dynamisch erzeugen und Zeiger einsammeln
var Left= 20; // Abstand vom linken Fensterrand in Pixeln
for (var i=0; i < 3; i++)
{
    Left+=10; // ab 30 Pixel vom linken Fensterrand
    document.write(
        '<DIV ID="' + DIV_ID + i.toString() + "' '
        + ' STYLE="position:absolute;'
        + 'left=' + Left + 'px;'
        + 'top=20px;'
        + '""
        + '>'
        + '</DIV>'
    );
    eval ('DIV_Zeiger' + i + ']=' + DIV_ID + i.toString() );
}

// DIV's dynamisch auf dem Bildschirm bezüglich dem linken Fensterrand um 20 Pixel
// verschieben
Left= 40;
for (i=0; i < 3; i++)
{
    Left+=10; // ab 50 Pixel vom linken Fensterrand verschieben
    eval('document.all.DIV_Zeiger' + i + ' .style.left=' + Left);
}
```

Hinweis: Anstelle von i.toString() kann auch nur i kodiert werden, da der Browser aufgrund von document.write() i automatisch nach String umwandelt. analog für i.toString() innerhalb eval()

Die Verschiebung erfolgt natürlich mit den Objekten, welche innerhalb von <DIV> .. </DIV> kodiert wurden. Im Falle von IMG's innerhalb des DIV werden diese mit verschoben.

Vor allem **wegen** der Manipulation von HTML-Objekten werden DIV's verwendet.

Kodierung der Style-Sheets-Angaben:

innerhalb CSS-Klasse:	immer mit Bindestrich
als .style-Eigenschaft:	ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabe

Bsp:	in CSS-Klasse	background-color
	in Script	.style.backgroundColor

Konvertierung:

Style besitzt für die style-gerechte Umwandlung eines Url-Wertes die Methode url(), welche nicht direkt als Methode des Style-Objektes implementiert ist und damit ohne Punktreferenz kodiert wird.




```

<STYLE TYPE="text/css">
<!--
    #freies_id{eigenschaften_liste} /* festkodiert ist #
//-->
</STYLE>
</HEAD>
<BODY>
    <tag_name ID="freies_id"> ..... </tag_name>
</BODY>

```

freies_id: Bsp: fettkursiv
 innerhalb STYLE mit vorgesetztem # kodieren !
 innerhalb BODY ohne # kodieren !

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

Variante 4 für CSS-Klasse, die einem HTML-Tag automatisch und ohne Verwendung des CLASS-Attributes zugeordnet wird, egal wo der Tag im BODY kodiert wurde:

```

<HEAD>
<STYLE TYPE="text/css">
<!--
    tag_namen_liste{eigenschaften_liste}
//-->
</STYLE>
</HEAD>
<BODY>
    <tag_name_1> ..... </tag_name_1>
    ....
    <tag_name_n> ..... </tag_name_n>
</BODY>

```

tag_namen_liste: mindestens 1 Element (Tag-Bezeichner)
 mehrere Elemente per Komma trennen
 alle Tags haben die gemeinsame Eigenschaftsliste
 Gross-klein bei Tag-Bezeichnern egal
 Bsp: h1,h2

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

Variante 5 für CSS-Klasse, die einem HTML-Tag mit Pseudoklasse automatisch und ohne Verwendung des CLASS-Attributes zugeordnet wird, egal wo der Tag im BODY kodiert wurde:

```

<HEAD>
<STYLE TYPE="text/css">
<!--
    tag_name:schlüsselwort{eigenschaften_liste} /* Doppelpunkt ist festkodiert
//-->
</STYLE>
</HEAD>
<BODY>
    <tag_name> ..... </tag_name>
</BODY>

```

tag_name: schlüsselwort ist Pseudoklasse zum Tag
 muss kodiert werden
 Bsp: a:link

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

Hinweise: Ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das tag-unabhängige ersetzt.
 Ein tag-unabhängiges Style-Sheet-Format fügt sich ansonsten zu den tag-abhängigen hinzu.



Variante 6 für CSS-Unterklasse, die im CLASS-Attribut zum entsprechenden HTML-Tag verwendet wird:

```
<HEAD>
  <STYLE TYPE="text/css">
    <!--
      tag_name.unterklasse_name{eigenschaften_liste}      /* Punkt ist festkodiert
    //-->
  </STYLE>
</HEAD>
<BODY>
  <tag_name CLASS="unterklasse_name"> ..... </tag_name>
</BODY>
```

tag_namen.unterklasse_name: Bsp: p.normal, muss aber im CLASS-Attribut zum Tag kodiert werden

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
Bsp: { font-weight:bold;font_style:italic;}

Hinweis: Aufgrund der Kodierung des CLASS-Attributes ist auch Variante 2 verwendbar. Warum Variante 6 existiert, ist unklar.

Beispiele für alle Varianten:

In den Beispielen wurden nur aus Gründen der Übersichtlichkeit Leerzeichen eingefügt.

Beispiel 1

```
<HEAD>
  <STYLE TYPE="text/css">
    <!--
      body{
        background-color:#FFFFFF;
        background-image:url(test.jpg);
        background-repeat:repeat-y;
        margin-top: 1cm;
        margin-left: 2cm;
        margin-right: 1cm;
      }
      p{
        font-family:serif;
        font-size: 12pt;
        text-align:justify;
      }
      p.zusatz_format_p_tag{text-indent:0.5cm;}
      .zusatz_format_alle_tags{
        position:absolute;
        top:4cm;
        z-index:3;
        font-size:40pt;
        font-weight:bold;
        color:blue;
        text-align:center;
      }
    -->
  </STYLE>
</HEAD>
<BODY>
  <SPAN CLASS="zusatz_format_alle_tags"> text </SPAN>
  <P CLASS="zusatz_format_p_tag"> text </P>
</BODY>
```

<!-- --> für Browser kodieren, die CSS nicht können

background-image: Hintergrundbild test.jpg verwenden
Achtung: Möglichst absolute Urls bzw. Pfadangaben verwenden, da eventuell Browser relative Angaben nicht interpretieren kann
Bsp.: relativ :url(ordner/test.gif)
absolut :url('www.test.de/ordner/test.gif')

background-repeat: repeat-y Hintergrundbild senkrecht wiederholen
repeat-x Hintergrundbild waagerecht wiederholen
no-repeat Hintergrundbild nicht wiederholen

p{....} gilt für JEDES P-Tag ohne CLASS-Attribut zu kodieren

margin-xxxx Seitenrand

text-align Textausrichtung

text-indent Einzug links

position Art der position-Angabe folgenden Angaben
Bsp: position:absolute --> Absolutangaben
top:4cm hier 4 cm



top	Abstand zum Browserfenster oben
z-index	Angezeigte Schicht (analog zu LAYER) --> ab 1 = unterste Schicht
p.zusatz_format_p_tag{ ..}	verwenden für überlappende Textbereiche per CLASS-Attribut anwenden im P-Tag

Beispiel 2

```

<HTML>
<HEAD>
  <STYLE TYPE="text/css">
  <!--
    .eigene_css_klasse{ .....}
  -->
  </STYLE>
</HEAD>
<BODY>
  <DIV CLASS=eigene_css_klasse>.....</DIV>
  .....
</BODY>
</HTML>

```

Style-Sheet-Deklaration per STYLE-Attribut im HTML-Element:

alle .style-Eigenschaft ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabenkodieren

Bsp:	in CSS-Klasse	background-color
	in Script	.style.backgroundColor

Beispiel:

```

<BODY>
  <tag_name STYLE=eigenschaften_liste>..... </tag_name>
</BODY>

```

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
Bsp: {font-weight:bold;font_style:italic;}

Style-Sheet-Deklaration in Script per Eigenschaft .style eines HTML-Elementes:

Beispiel:

```

<DIV ID="ID_Div">Test</DIV>
document.ID_Div.style.height="200px";

```

4.3.2.2.4.3.39.2. Style-Sheet und vordefinierte Werte**Style-Sheet und Farbangaben**

browserspezifisch

Bsp.: "yellow"

Hinweis: frei wählbare Farben per RGB-Angaben im Format

```

#rrggbb
rgb(rrr,ggg,bbb)

```

Rot-Anteil:

hexa	rr	0 bis FF, mit Vornull
dezimal 0-255	rrr	ohne Vornullen

Grün-Anteil:

hexa	gg	0 bis FF, mit Vornull
dezimal 0-255	ggg	ohne Vornullen

Blau-Anteil:

hexa	bb	0 bis FF, mit Vornull
dezimal 0-255	bbb	ohne Vornullen

Beispiele: #FF01A3

rgb(1,255,10)

Style-Sheet und Dimensionen

pt	Point	=	1/72 inches
pc	Pica	=	12 pt
in	Inch	=	2,54 cm
mm			
cm			
px	Pixel		
em	relativ zur Schrifthöhe des Elementes, das per CSS formatiert wird		
ex	relativ zur Höhe des Buchstaben Grosses X		
%	Prozentanteil von der Norm des per CSS formatierten Elementes		

Style-Sheet- und Dezimalkomma bei numerischen Werten

Dezimalkomma ist der Punkt und NICHT das Komma.

(TWS) Microsoft JScript für den Hobby-Programmierer



Style-Sheet und Schriften (Font und Style)

vordefinierte Schriftarten:

- serif
- san-serif
- cursive
- fantasy
- monospace

vordefinierte Style:

- italic entspricht kursiv
- oblique entspricht kursiv
- normal entspricht nicht kursiv

4.3.2.2.4.3.39.3. *Style-Sheet aus Datei einbinden***Style-Sheet-Schriftdatei laden (*.eot bzw. *.prf)**

Microsoft: *.eot

Netscape: *.prf

```
<STYLE TYPE="text/css">
<!--
    @font-face{ eigenschaften_liste; src:url (eot_bzw._prf_datei_liste);}
// -->
</STYLE>
```

eigenschaften_liste: optional
 Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

eot_bzw._prf_datei_liste: mindestens 1 Element
 Kommatrennung der Elemente

lokale Fontdateien laden

Fontdateien liegen im FONT-Ordner von Windows

```
<STYLE TYPE="text/css">
<!--
    @font-face{ eigenschaften_liste; local (font_datei_liste);}
oder
    @font-face{ eigenschaften_liste; format (TrueType);}
// -->
</STYLE>
```

eigenschaften_liste: optional
 Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

font_datei_liste: lokale Fontdateien im FONT-Ordner von Windows
 mindestens 1 Element
 Blanktrennung der Elemente

TrueType es werden nur lokale TrueTyp-Fonts (ttf-Dateien) aus dem FONT-Ordner von Windows verwendet

Style-Sheet-Datei laden (*.css)

per LINK-Tag:

```
<LINK
REL=stylesheet
TYPE="text/css"
HREF="url_oder_css_dateiname"
[MEDIA="typ_name"]
>
<STYLE TYPE="text/css">
<!-- weitere Stylesheet-Angaben
// -->
</STYLE>
```

MEDIA= optional
 typ_name: Name des Ausgabe-Mediums
 "all" alle Medien



"screen"	Bildschirm
"print"	Drucker

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x und NS 6.x):

```
<BODY>
  <LINK REL="stylesheet" MEDIA="print" HREF="print.css">
  <DIV CLASS="keindruck">
    alle Seitenteile, die nicht gedruckt werden sollen
  </DIV>
</BODY>

print.css enthält nur
.keindruck {display:none; }
```

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

```
<BODY>
  <LINK REL="stylesheet" MEDIA="screen" HREF="screen.css">
  <DIV CLASS="nurdruck">
    alle Seitenteile, die nur auf Ausdruck zu sehen sein sollen
  </DIV>
</BODY>
```

per Pseudoklasse @import:

```
<STYLE TYPE="text/css">
<!--
  @import (url_oder_css_datei) typ_liste;
// -->
</STYLE>

typ_liste: Liste der Ausgabe-Medien
           Kommatrennung
           Ausgabe-Medien können sein
             "all"           alle Medien
             "screen"        Bildschirm
             "print"         Drucker
           Bsp: print,screen;
```

4.3.2.2.4.3.39.4. Style-Sheet und Wertzuweisung an Eigenschaften

eigenschaft: wert; oder eigenschaft=wert;

Hinweis: Pflichtkodierung von Doppelpunkt bzw. Gleichheitszeichen
Semikolon

wenn für wert auto kodiert werden kann, so ist das der Standardwert, falls Style nicht belegt wird

Folge von Eigenschaften möglich: als Liste mit Semikolontrennung

Bsp: eigenschaft1=wert1;; eigenschaft_n=wert_n;

wenn Blank in Kodierung enthalten, so alles in " " bzw. ' ' setzen

Bsp: style="font-style: italic; color:red;"

wenn Eigenschaft mit mehreren Werten belegbar, so

Werte durch Blank trennen

gesamte Liste in " " bzw. ' ' kodieren wegen den Trennblanks

Bsp: "background: url(test.gif) no-repeat middle;"

4.3.2.2.4.3.39.5. Style-Sheet und Ausgabemedien (Pseudoklasse @media)

```
<STYLE TYPE="text/css">
<!--
  @media typ_liste{eigenschaften_liste}
// -->
</STYLE>

typ_liste: Liste der Ausgabe-Medien
           Kommatrennung
           Ausgabe-Medien können sein
             "all"           alle Medien
             "screen"        Bildschirm
             "print"         Drucker
           Bsp: print,screen;
```



eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " bzw. ' ' setzen
 Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.6. **Style-Sheet und Seiten-Eigenschaften (Pseudoklasse (@page))**

@page:first {eigenschaften_liste} Regel gehört der 1. Seite
 @page:footer{eigenschaften_liste} Regel gehört der Fußnote
 @page:header{eigenschaften_liste} Regel gehört der Kopfnote
 @page:left{eigenschaften_liste} Regel gehört der linken Seite
 @page:left:header{eigenschaften_liste} Regel gehört der Kopfnote Seite links
 @page:left:footer{eigenschaften_liste} Regel gehört der Fußnote Seite links
 @page:right{eigenschaften_liste} Regel gehört der rechten Seite
 @page:right:header{eigenschaften_liste} Regel gehört der Kopfnote Seite rechts
 @page:right:footer{eigenschaften_liste} Regel gehört der Fußnote Seite rechts

Beispiel: @page : left {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.7. **Style-Sheet und HTML-Tag-bezogene Eigenschaften**

Hinweis: Ist für einen Style der Wert auto kodierbar, so ist das der Standardwert, falls Style nicht belegt wird.

Style-Sheet-Eigenschaften zu <A> und seine Pseudoklassen

a:link{eigenschaften_liste} Eigenschaften für noch nicht besuchte Links
 a:visited {eigenschaften_liste} Eigenschaften für schon besuchte Links
 a:active{eigenschaften_liste} Eigenschaften für gerade besuchtes Link
 a:hover{eigenschaften_liste} Eigenschaften für Maus über Link , ab IE 4.x
 nur für Tags mit HREF-Attribut

Style-Sheet-Eigenschaften zu <P> und seine Pseudoklassen

p:first-line{eigenschaften_liste} Eigenschaften Absatz 1. Zeile
 p:first-letter{eigenschaften_liste} Eigenschaften Absatz 1. Zeile, 1. Zeichen
 p:before{content:"freier_text" } Text vor dem Element einfügen
 p:after{content:"freier_text" } Text nach dem Element einfügen

Style-Sheet-Eigenschaften zu <H1> bis <H6>

h1:first-line{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile
 h1:first-letter{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile, 1. Zeichen

für H2 bis H6 analog

Beim Internet Explorer 6.0 unter Windows 98 bzw. Windows XP wurde die Darstellung der Überschriften verändert, wenn der jeweilige Standardfont benutzt wird. Empfehlung: Neudefinition der <Hx>-Tags per Style (soweit möglich).

Style-Sheet-Eigenschaften für Tabelle

caption-side:top; Überschrift oberhalb zentriert
 caption-side:topleft; Überschrift oberhalb linksbündig
 caption-side:topright; Überschrift oberhalb rechtsbündig
 caption-side:bottom; Überschrift unterhalb zentriert
 caption-side:bottomleft; Überschrift unterhalb linksbündig
 caption-side:bottomright; Überschrift unterhalb rechtsbündig

row-span: anzahl_der_zeilen_über_die_sich_die_zelle_ausstrecken_soll;
 column-span: anzahl_der_spalten_über_die_sich_die_zelle_ausstrecken_soll;

Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet)

display: list-item; Element in eigenem Absatz UND mit Bullet anzeigen

Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung)

list-style-type:decimal;	1	2	3 ...
list-style-type:lower-roman;	i	ii	iii ..
list-style-type:lower-alpha	a	b	c
list-style-type:upper-roman;	I	II	III ...
list-style-type:upper-alpha;	A	B	C
list-style-type:disk;	Bullet ist Dateisymbol		
list-style-type:circle;	Bullet ist rund		
list-style-type:square;	Bullet ist rechteckig		
list-style-type:none;			

list-style-position:inside; Listenelement einrücken; ist Standard
 list-style-position:outside Listenelement ausrücken

list-style-image: url(bullet_grafik_datei); GIF oder JPG

list-style: liste_aller_obigen_eigenschaften_mit_blank_trennung;



Style-Sheet und Dimension von Elementen

height:wert;	Höhe des Objektes
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe Hinweis: wenn geändert wird, so kann automatischer Umbruch des Inhaltes des HTML-Elementes erfolgen
height:auto;	
width:wert;	Breite des Objektes
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe Hinweis: wenn geändert wird, so kann automatischer Umbruch des Inhaltes des HTML-Elementes erfolgen
width:auto;	

Style-Sheet und Position von Elementen

position: absolute;	Positionsangaben bezüglich Anzeigefenster-Rand, Element ist scrollbar
position: fixed;	Positionsangaben bezüglich Anzeigefenster-Rand, Element ist nicht scrollbar
position: relative;	Positionsangaben bezüglich Vorgänger-Element
position: static;	hebt absolute bzw. fixed bzw. relative auf
top:wert;	Abstand zum oben umgebenden Objekt ab IE 5.x
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
top:auto;	
bottom:wert;	Abstand zum unten umgebenden Objekt ab IE 5.x
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
bottom:auto;	
left:wert;	Abstand zum links umgebenden Objekt ab IE 5.x
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
left:auto;	
right:wert;	Abstand zum rechts umgebenden Objekt ab IE 5.x
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe
right:auto;	
offset-left:	wie left ab IE 5.x
offse-top:	wie top ab IE 5.x
pixel-left:wert;	absolute X-Position der linken oberen Objekt-Ecke im Pixelsystem, dessen Ursprung (0,0) links oben liegt, also Abstand vom linken Fensterrand wert nur Integer und ohne Einheit, da automatisch in Pixel interpretiert
pixel-top:wert;	absolute Y-Position der linken oberen Objekt-Ecke im Pixelsystem, dessen Ursprung (0,0) links oben liegt, also Abstand vom oberen Fensterrand wert nur Integer und ohne Einheit, da automatisch in Pixel interpretiert

Style-Sheet und Abstand von HTML-Elementen

top:abstand_oberhalb_in_pixel;	
top:auto;	
left:abstand_links_in_pixel;	
left:auto;	
bottom:abstand_unterhalb_in_pixel;	
bottom:auto;	
right:abstand_rechts_in_pixel;	
right:auto;	
margin: werte_liste_von_breiten_mit_blank_trennung;	Randbreite aller 4 Seiten
wert ist	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten 2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für links UND rechts



	3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für rechts 3. Wert gilt für unten 4. Wert gilt für links
margin:auto;	
margin-top: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite oben wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-top:auto;	
margin-bottom: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite unten wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-bottom:auto;	
margin-left: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite links wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-left:auto;	
margin-right: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite rechts wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-right:auto;	
padding: werte_liste_von_breiten_mit_blank_trennung;	Innenabstand aller 4 Seiten
wert ist	Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten 2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für links UND rechts 3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für rechts 3. Wert gilt für unten 4. Wert gilt für links
padding:auto;	
padding-top: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand oben wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-top:auto;	
padding-bottom: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand unten wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-bottom:auto;	
padding-left: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand links wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-left:auto;	
padding-right: werte_liste_obiger_breiten_mit_blank_trennung;	Innenabstand rechts wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
padding-right:auto;	

Style-Sheet und Überlagerung von Elementen

z-index:auto;	jedes neu erzeugte Objekt ist das oberstes
Achtung:	Sollte die Dimension des neuen Objektes sich nach der Erzeugung derart verändern, so dass es ein anderes Objekt überlagert, dann ändert sich trotzdem nichts an der Sichtbarkeit
z-index:wert;	Schichtnummer des Objektes bei sich irgendwann überlagernden Objekten (also auch bei Überlagerung, die erst nach deren Erzeugung erfolgt)
Wert	Integer ganzzahlig, also auch < 0 je kleiner um so tiefer in der Schicht je höher um so höher in der Schicht Sichtbar ist immer die oberste, also höchste Schicht wenn 2 Objekte mit identischer Schichtnummer, so Sichtbarkeit laut Reihenfolge der Kodierung im Quelltext



Style-Sheet und HTML-Elemente-Anordnung im Dokument

direction:ltr;	von links nach rechts; ist Standard
direction:rtl;	von rechts nach links
display:block;	Element in eigenem Absatz anzeigen
display:inline;	Element nicht in eigenem Absatz anzeigen
display:none;	Element nicht anzeigen
display:list-item;	
float:left;	Element linksbündig; Textfluss rechts
float:right;	Element rechtsbündig; Textfluss links
float:none;	Standard
clear:left;	hebt float:left; auf
clear:right;	hebt float:right; auf
clear:both;	hebt float:left; UND float:right; auf
clear:none;	identisch mit float:none;

Style-Sheet und HTML-Element-Dimension (-Grenzen, -Anzeigebereich)

width: breite_in_pixel;	
width:auto;	
min-width: minimale_breite_in_pixel;	
min-width:auto;	
max-width: maximale_breite_in_pixel;	
max-width:auto;	
height: hoehe_in_pixel;	
height:auto;	
min-height: minimale_hoehe_in_pixel;	
min-height:auto;	
max-height: maximale_hoehe_in_pixel;	
max-height:auto;	
overflow:hidden;	wenn Elementgröße > max-width und max-height, so wird Element auf Maximalwerte begrenzt
overflow:visible;	wenn Elementgröße > max-width und max-height, so werden Maximalwerte ignoriert

Style-Sheet und HTML-Element-Sichtbarkeit

visibility:hidden;	unsichtbar, aber Platzhalter vorhanden, so dass das HTML-Element im Layout unsichtbar bleibt
visibility:visible;	sichtbar
visibility:inherit;	Sichtbarkeit laut Elternobjekt
display:none;	unsichtbar UND kein Platzhalter möglich, also HTML-Element auch aus dem Layout entfernen
display:block;	sichtbar
display:inline;	sichtbar

Style-Sheet und Hintergrund-Grafik

background-image:url (grafik_datei);	Hintergrunddatei GIF oder JPG
background-image:none;	
background-color:#rrggbb;	Hintergrundfarbe
background-color:rgb(rrr,ggg,bbb);	
background-color:vordefinierte_farbe;	
background-repeat:repeat;	Hintergrundgrafik kacheln auf gesamten Hintergrund
background-repeat:repeat-x;	Hintergrundgrafik für 1 Zeile kacheln
background-repeat:repeat-y;	Hintergrundgrafik für 1 Spalte kacheln
background-repeat:no-repeat;	kein Kacheln des Hintergrundbildes
background-attachment:scroll;	Hintergrundgrafik scrollt mit Vordergrund
background-attachment:fixed;	Hintergrundgrafik scrollt nie
background-position:top;	Hintergrundgrafik auf Oberkante Hintergrund
background-position:center;	Hintergrundgrafik zentriert auf Hintergrund
background-position:middle;	Hintergrundgrafik mittig auf Hintergrund
background-position:bottom;	Hintergrundgrafik auf Unterkante Hintergrund
background-position:left;	Hintergrundgrafik linksbündig auf Hintergrund
background-position:right;	Hintergrundgrafik rechtsbündig auf Hintergrund

background: eigenschaften_liste_mit_blank_trennung; alle obigen Eigenschaften kodierbar mit Blanktrennung

Bsp: background: url(back.gif) no-repeat middle; Blank-Trennung !!

-moz-opacity:faktor Transparenz des Hintergrundbildes
nur NS 6.x



Faktor > 0 und bis 1

Bsp: 0.9 entspricht 90%

Bsp:

Style-Sheet und Rahmen-Eigenschaften (Border)

border-color:#rrggbb;
border-color:rgb(rrr,ggg,bbb);
border-color:vordefinierte_farbe;
border-top-color:#rrggbb;
border-top-color:rgb(rrr,ggg,bbb);
border-top-color:vordefinierte_farbe;
border-bottom-color:#rrggbb;
border-bottom-color:rgb(rrr,ggg,bbb);
border-bottom-color:vordefinierte_farbe;
border-left-color:#rrggbb;
border-left-color:rgb(rrr,ggg,bbb);
border-left-color:vordefinierte_farbe;
border-right-color:#rrggbb;
border-right-color:rgb(rrr,ggg,bbb);
border-right-color:vordefinierte_farbe;

Rahmenfarbe aller 4 Seiten

Rahmenfarbe oben

Rahmenfarbe unten

Rahmenfarbe links

Rahmenfarbe rechts

border-style:none;
border-style:dotted;
border-style:dashed;
border-style:solid;
border-style:double;
border-style:groove;
border-style:ridge;
border-style:inset;
border-style:outset;
border-top-style:none;
border-top-style:dotted;
border-top-style:dashed;
border-top-style:solid;
border-top-style:double;
border-top-style:groove;
border-top-style:ridge;
border-top-style:inset;
border-top-style:outset;
border-bottom-style:none;
border-bottom-style:dotted;
border-bottom-style:dashed;
border-bottom-style:solid;
border-bottom-style:double;
border-bottom-style:groove;
border-bottom-style:ridge;
border-bottom-style:inset;
border-bottom-style:outset;
border-left-style:none;
border-left-style:dotted;
border-left-style:dashed;
border-left-style:solid;
border-left-style:double;
border-left-style:groove;
border-left-style:ridge;
border-left-style:inset;
border-left-style:outset;
border-right-style:none;
border-right-style:dotted;
border-right-style:dashed;
border-right-style:solid;
border-right-style:double;
border-right-style:groove;
border-right-style:ridge;
border-right-style:inset;
border-right-style:outset;

kein Rahmen um das gesamte Element
gepunkteter Rahmen um das gesamte Element
gestrichelter Rahmen um das gesamte Element
einfacher durchgezogener Rahmen um das gesamte Element
doppelter durchgezogener Rahmen um das gesamte Element
3D-Effekt 1 um das gesamte Element
3D-Effekt 2 um das gesamte Element
3D-Effekt 3 um das gesamte Element
3D-Effekt 4 um das gesamte Element
kein Rahmen oben
gepunkteter Rahmen oben
gestrichelter Rahmen oben
einfacher durchgezogener Rahmen oben
doppelter durchgezogener Rahmen oben
3D-Effekt 1 oben
3D-Effekt 2 oben
3D-Effekt 3 oben
3D-Effekt 4 oben
kein Rahmen unten
gepunkteter Rahmen unten
gestrichelter Rahmen unten
einfacher durchgezogener Rahmen unten
doppelter durchgezogener Rahmen unten
3D-Effekt 1 unten
3D-Effekt 2 unten
3D-Effekt 3 unten
3D-Effekt 4 unten
kein Rahmen links
gepunkteter Rahmen links
gestrichelter Rahmen links
einfacher durchgezogener Rahmen links
doppelter durchgezogener Rahmen links
3D-Effekt 1 links
3D-Effekt 2 links
3D-Effekt 3 links
3D-Effekt 4 links
kein Rahmen rechts
gepunkteter Rahmen rechts
gestrichelter Rahmen rechts
einfacher durchgezogener Rahmen rechts
doppelter durchgezogener Rahmen rechts
3D-Effekt 1 rechts
3D-Effekt 2 rechts
3D-Effekt 3 rechts
3D-Effekt 4 rechts

border-width:wert

Rahmendicke aller 4 Seiten

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm

border-width:medium

mittel

border-width:thin

dünn

border-width:thick

dick

border-top-width:wert
oder cm

Rahmendicke oben, wert ist Fließkommazahl mit Einheitenangabe z.B. pt



border-top-width:medium	mittel oben
border-top-width:thin	dünn oben
border-top-width:thick	dick oben
border-bottom-width:wert	Rahmendicke unten, wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
border-bottom-width:medium	mittel unten
border-bottom-width:thin	dünn unten
border-bottom-width:thick	dick unten
border-left-width:wert oder cm	Rahmendicke links, wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
border-left-width:medium	mittel links
border-left-width:thin	dünn links
border-left-width:thick	dick links
border-right-width:wert	Rahmendicke rechts, wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
border-right-width:medium	mittel rechts
border-right-width:thin	dünn rechts
border-right-width:thick	dick rechts
border-top:eigenschaften_liste;	passende Werte-Liste für Rahmen oben, Werte mit Blank trennen
border-bottom: eigenschaften_liste;	passende Werte-Liste für Rahmen unten, Werte mit Blank trennen
border-right: eigenschaften_liste;	passende Werte-Liste für Rahmen rechts, Werte mit Blank trennen
border-left: eigenschaften_liste;	passende Werte-Liste für Rahmen links, Werte mit Blank trennen
border:eigenschaften_liste;	Werte-Liste aller möglichen Bordereigenschaften, Werte mit Blank trennen
Bsp: border-top: thick inset rgb(192,192,255);	

Style-Sheet und Element-Ausschnitt

clip: rect (x1,y1,x2,y2);

Ausschnitt
 Pixelangaben bezüglich Elementgrenze
 Form des Bildausschnittes: RECHTECKIG
 wird eigentlich nur zum Schreiben verwendet
 x1,y1 linke obere Ecke
 x1 horizontal in Pixel
 y1 vertikal in Pixel
 x2,y2 rechte untere Ecke
 x2 horizontal in Pixel
 y2 vertikal in Pixel
 x1 bis y2 immer bezüglich links oben im HTML-Element
 x1,y1, x2,y2 können den Wert auto haben, der immer die maximale Dimension bewirkt
 Bsp. 'rect(auto,y1,auto,y2)';
 wenn alle auf auto, so wird der Bildausschnitt maximiert, was der Standarddimension des Elementes entspricht (kein Ausschnitt)

clip:auto; kein Ausschnitt
 entspricht clip:rect(auto,auto,auto,auto);

Style-Sheet und Element-Scrolling

Scrolling nötig, wenn

Elementgröße > max-width und max-height
 aber Maximalwerte eingehalten werden sollen

overflow:scroll; Scrolling immer möglich
 overflow:auto;

siehe auch overflow:visible; bzw. overflow:hidden;

Style-Sheet-Eigenschaften für Seitendarstellung im Dokument

margin: werte_liste_von_breiten_mit_blank_trennung; Randbreite aller 4 Seiten

wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 Prozentangabe z.B. 3%

wenn nur 1 Wert kodiert: gilt für oben UND unten UND links UND rechts

wenn 2 Werte kodiert: 1. Wert gilt für oben UND unten
 2. Wert gilt für links UND rechts

wenn 3 Werte kodiert: 1. Wert gilt für oben
 2. Wert gilt für links UND rechts
 3. Wert gilt für unten

wenn 4 Werte kodiert: 1. Wert gilt für oben
 2. Wert gilt für rechts
 3. Wert gilt für unten
 4. Wert gilt für links



margin:auto;	
margin-top: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite oben wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-top:auto;	
margin-bottom: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite unten wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-bottom:auto;	
margin-left: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite links wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-left:auto;	
margin-right: werte_liste_obiger_breiten_mit_blank_trennung;	Randbreite rechts wert ist Fließkommazahl mit Einheitenangabe z.B. pt oder cm Prozentangabe z.B. 3%
margin-right:auto;	
size:seiten_breite;	z.B. 20cm
size:seiten_hoehe;	z.B. 29cm
size:seiten_breite seiten_hoehe;	Blanktrennung der beiden Werte
size:landscape;	Querformat
size:portrait;	Hochformat
size:auto;	
page-break_before:always;	immer Seitenumbruch vor aktuellem Element erzeugen
page-break_before:aroid;	nie Seitenumbruch vor aktuellem Element erzeugen
page-break_before:left;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann linksbündig ablegen
page-break_before:right;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann rechtsbündig ablegen
page-break_before:auto;	
page-break_after:always;	immer Seitenumbruch nach aktuellem Element erzeugen
page-break_after:aroid;	nie Seitenumbruch nach aktuellem Element erzeugen
page-break_after:left;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element linksbündig ablegen
page-break_after:right;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element rechtsbündig ablegen
page-break_after:auto;	

Style-Sheet-Eigenschaften für Text

text-indent:wert;	Texteinschub wert ist Fließkommazahl mit Einheit z.B. mm oder cm oder pt Prozent z.B. 3% wert > 0, so Textzeile einrücken wert < 0, so Textzeile ausrücken
text-align:left;	Text linksbündig
text-align:center;	Text zentriert
text-align:right;	Text rechtsbündig
text-align:justify;	Text im Blocksatz
vertical-align:top;	Text auf Oberkante der Umgebung
vertical-align:middle;	Text auf Mitte der Umgebung
vertical-align:bottom;	Text auf Unterkante der Umgebung
vertical-align:sub;	Text tieferstellen
vertical-align:super;	Text höher stellen
vertical-align:baseline;	Text auf Basislinie des umgebenden Textes mit verschiedenen Schriftarten
vertical-align:text-top;	Text auf obere Linie des umgebenden Textes mit verschiedenen Schriftarten
vertical-align:text-bottom;	Text auf untere Linie des umgebenden Textes mit verschiedenen Schriftarten
font-size:schriftgroesse_wert;	z.B. 2pt oder vordefiniert
line-height:zeilenhoehe_wert;	z.B. 2pt* oder vordefiniert
white-space:normal;	automatischer Zeilenumbruch einschalten, ab IE 5.x
white-space:pre;	manuellen Zeilenumbruch einschalten, ab IE 5.x
white-space:nowrap;	kein Zeilenumbruch möglich, ab IE 5.x
color:#rrggbb;	
color:rgb(rrr,ggg,bbb);	
color:vordefinierte_farbe;	
columns:anzahl_der_text_spalten_fuer_textfluss;	
column-gap:abstand_der_text_spalten_fuer_textfluss;	



column-rule-width:dicke_des_trennstriches_zwischen_den_textspalten_vom_textfluss;
column-rule-color:#rrggbb; oder rgb(rrr,ggg,bbb); Farbe Trennstrich
column-rule-style:none; kein Trennstrich zwischen Textspalten
column-rule-style:dotted; gepunkteter Trennstrich
column-rule-style:dashed; gestrichelter Trennstrich
column-rule-style:solid; einfacher durchgezogener Trennstrich
column-rule-style:double; doppelter durchgezogener Trennstrich
column-rule-style:groove; 3D-Effekt 1
column-rule-style:ridge; 3D-Effekt 2
column-rule-style:inset; 3D-Effekt 3
column-rule-style:outset; 3D-Effekt 4
column-rule:werte_liste_aus_obigen_textfluss_werten_mit_blank_trennung;

word-spacing: abstand_zwischen_woertern_im_text;

word-wrap:normal; kein Wortumbruch, ab IE 5.x
word-wrap:break-word; Wortumbruch, ab IE 5.x

letter-spacing:abstand_zwischen_zeichen_im_text; Fließkommazahl mit Einheit z.B. mm oder cm oder pt
letter-spacing:normal;

line-height:abstand_zwischen_2_zeilen; Fließkommazahl mit Einheit z.B. mm oder cm oder pt
line-height:normal;

text-decoration:underline; Text unterstrichen
text-decoration:overline; Text überstrichen
text-decoration:line-through; Text durchgestrichen
text-decoration:blink; Text blinkend
text-decoration:none; Text normal

text-transform:capitalize; Wortanfang ALLER Wörter auf Großbuchstabe
text-transform:uppercase; alle Zeichen nach Großbuchstabe
text-transform:lowercase; alle Zeichen nach Kleinbuchstabe
text-transform:none; Text normal

text-shadow:#rrggbb;
text-shadow:rgb(rrr,ggg,bbb);
text-shadow:vordefinierte_farbe;
text-shadow:none; kein Textschatten

orphans:anzahl_der_VOR_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
entspricht Schusterjunge

widow:anzahl_der_NACH_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
entspricht Hurenkind

Style-Sheet-Eigenschaften für Unicode (Zeichensatz)

unicode-range:U+xxxx-yyy;

Fragezeichen als Joker innerhalb von xxxx und yyyy verwendbar

Bsp: unicode-range:U+0000-007F; ist ASCII-Zeichensatz
 unicode-range:U+0000-00?F; ? steht für 0 bis F --> ist nicht ASCII-Zeichensatz

Style-Sheet-Eigenschaften für Font

font-family: schriftarten_liste_mit_kommatrennung;

es wird **nur** der **ERSTE** auf dem lokalen Rechner **gefundene** Font aus der Liste geladen
wenn Blank vorhanden, so alles in " " bzw. ' ' setzen
vordefinierte Schriftarten:

serif
san-serif
cursive
fantasy
monospace

font-style:italic; kursiv
font-style:oblique; kursiv
font-style:normal; nicht kursiv

font-variant:small-caps; kleine Großbuchstaben (Kapitälchen)
font-variant:normal; kein Kapitälchen

font-size:schrift_groesse_wert; Fließkommazahl mit Einheit z.B. mm oder cm oder pt
Prozent z.B. 3%
font-size:xx-small; winzig



font-size:x-small;	sehr klein
font-size:small;	klein
font-size:medium;	mittel
font-size:large;	groß
font-size:x-large;	sehr groß
font-size:xx-large;	riesig
font-size:smaller;	etwas kleiner als normal
font-size:larger;	etwas größer als normal

font-weight:bold;	fett
font-weight:bolder;	extra fett
font-weight:lighter;	dünn
font-weight:normal;	nicht dünn und nicht fett
font-weight:100;	extra dünn
font-weight:200;	
font-weight:300;	
font-weight:400;	
font-weight:500;	medium
font-weight:600;	
font-weight:700;	bold
font-weight:800;	
font-weight:900;	extra fett

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;

Liste darf maximal 6 der nachfolgenden Eigenschaften beinhalten

caption	Zeichensatz für Objekte mit Überschriften
font-style	
font-variant	
font-weight	
font-size	
line-height	
font-family	
icon	Zeichensatz für Grafik
menu	Zeichensatz in Menüs
message-box	Zeichensatz in Messages-Boxen
small-caption	Zeichensatz für Control-Elemente
status-bar	Zeichensatz für Statuszeile

Style-Sheet-Eigenschaften für Mauscursor

cursor:auto;	
cursor:default;	je nach Windowseinstellung
cursor:hand;	
cursor:crosshair;	Fadenkreuz
cursor:pointer;	Zeiger
cursor:move;	Kreuz für Beweglichkeit
cursor:n-resize;	Pfeil Nord
cursor:ne-resize;	Pfeil Nord-Ost
cursor:nw-resize;	Pfeil Nord-West
cursor:e-resize;	Pfeil Ost
cursor:se-resize;	Pfeil Süd-Ost
cursor:s-resize;	Pfeil Süd
cursor:sw-resize;	Pfeil Süd-West
cursor:w-resize;	Pfeil West
cursor:text;	also senkrechter Strich
cursor:wait;	Sanduhr
cursor:help;	Fragezeichen
cursor:url(url_oder_mauscursor_grafik_datei);	GIF oder JPG

Style-Sheet-Eigenschaften für Scrollbalken (Scrollbars)

scrollbar3d-light-color:#rrggbb	Scrollbar-Farbe bei 3D
scrollbar3d-light-color:rgb(rrr,ggg,bbb);	
scrollbar3d-light-color:vordefinierte_farbe;	
scrollbar-arrow-color:#rrggbb;	Scrollbar-Pfeile-Farbe ab IE 5.x
scrollbar-arrow-color:rgb(rrr,ggg,bbb);	
scrollbar-arrow-color:vordefinierte_farbe;	
scrollbar-base-color:#rrggbb;	Scrollbar-Basis-Farbe ab IE 5.x
scrollbar-base-color:rgb(rrr,ggg,bbb);	
scrollbar-base-color:vordefinierte_farbe;	
scrollbar-dark-shadow-color:#rrggbb;	Scrollbar-Farbe des dunklen Schattens ab IE 5.x
scrollbar-dark-shadow-color:rgb(rrr,ggg,bbb);	
scrollbar-dark-shadow-color:vordefinierte_farbe;	
scrollbar-face-color:#rrggbb;	Scrollbar-Face-Farbe ab IE 5.x



```

scrollbar-face-color:rgb(rrr,ggg,bbb);
scrollbar-face-color:vordefinierte_farbe;
scrollbar-highlight-color:#rrggbb;
scrollbar-highlight-color:rgb(rrr,ggg,bbb);
scrollbar-highlight-color:vordefinierte_farbe;
scrollbar-shadow-color:#rrggbb;
scrollbar-shadow-color:rgb(rrr,ggg,bbb);
scrollbar-shadow-color:vordefinierte_farbe;

```

Scrollbar-Highlight-Farbe ab IE 5.x

Scrollbar-Farbe des Schattens ab IE 5.x

Style-Sheet-Eigenschaften für Zoom eines Elementes

```

zoom:wert;
zoom:normal;

```

Objekt vergrößern ab IE 5.x
wert ist Fließkommazahl als Faktor (1,0 ist normal)
Prozentangabe mit % z.B. 50% (100% ist normal)

4.3.2.2.4.3.39.8. Style-Sheet-Beispiele

Beispiel 1

```

<STYLE TYPE="text/css">
<!--
    h1 {font-size:24pt;margin-top:1.2cm;margin-left:30px;}
    body {background-color:rgb(51,0,102);}
    p.li {font-size:12pt;line-height:14pt;font-family:Helvetica,Arial;}
    p.normal {font-size:10pt;color:black;}
    p.klein {font-size:8pt;color:black}
    all.rot {color:red;}    oder .rot {color:red}
                                Rot-Definition für ALLE Tags
                                Anwendung z.B. per <P CLASS="normal">text</P>
                                Anwendung z.B. per <P CLASS="rot">text</P>
                                Anwendung z.B. per <H1 CLASS="rot">text</H1>
    #fett_kursiv {font-weight:bold;font-style:italic;}
                                Anwendung z.B. per <P ID="fett_kursiv">text</P>
    a:link {color:#FF0000;font-weight:bold;}
                                Anwendung z.B. per <A HREF=".....">text</A>
// -->
</STYLE>

```

Anwendung je nach Tag --> meist innerhalb <BODY>

Beispiel 2

```

<BODY>
    <DIV STYLE="background-color:#FF0000;">text</DIV>
    <P>
        text1
        <SPAN STYLE="color:red;">
            text2_in_rot;
        </SPAN>
        text3_wie_text1
    </P>
</BODY>

```

Aufgrund der HTML-Integration von CSS sollte der Browser CSS unterstützen. Ob diese Unterstützung gegenüber den aktuellen Standards zu HTML und CSS vollständig ist, hängt vom Willen des Browserherstellers ab. Letztendlich wird CSS sinnigerweise als grundlegende Komponente und Eigenschaft **aller** Objekte integriert. CSS bietet elementare Vielfalt bei der dynamischen Scriptprogrammierung, ist gewöhnungsbedürftig, aber durch Normung universell auf diverse Objekte anwendbar, die das STYLE-Attribut bzw. die Eigenschaft .style unterstützen.

Es sei darauf hingewiesen, dass

die Verwendung von CSS-Klassen, Kodierung von <STYLE> bzw. dem STYLE-Attribut bzw. der Eigenschaft .style nur **scheinbar** alle synonym sind:

CSS-Klassen im HEAD können dokumentweit verwendet werden.

<STYLE>, STYLE-Attribut und .style sind stets HTML-komponentenbezogen.

der Browser diese Formen intern z.T. verschieden (auch bezüglich der Collectionen) verwaltet.

Das Einbinden von Styles aus einer externen Datei ist möglich.

Nachfolgende Beschreibungen orientieren sich am STYLE-Attribut, sind aber auf alle o.g. Formen sinngemäß übertragbar.

Hinweis zu Style-Begriffen:

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objekthinhaltes zum Aussenrand

Beispiel: Es sollen DIVs erzeugt werden und deren Objektreferenzen (Zeiger) in einem Zeigerfeld gesammelt werden.

Das Zeigerfeld dient der vereinfachten Verwaltung der dynamisch erzeugten DIV's.



```

var DIV_ID="Test_DIV"; // auch "Otto_DIV" etc.möglich , je nach Wunsch

var DIV_Zeiger=Array(); // sammelt alle ID als Zeichenketten

// DIV's dynamisch erzeugen und Zeiger einsammeln
var Left= 20; // Abstand vom linken Fensterrand in Pixeln
for (var i=0; i < 3; i++)
{
    Left+=10; // ab 30 Pixel vom linken Fensterrand
    document.write(
        '<DIV ID="' + DIV_ID + i.toString() + "' '
        + ' STYLE="position:absolute;'
        + ' left=' + Left + 'px;'
        + ' top=20px;'
        + ' "'
        + '>'
        + '</DIV>'
    );
    eval ('DIV_Zeiger' + i + ']=' + DIV_ID + i.toString() );
}

// DIV's dynamisch auf dem Bildschirm bezüglich dem linken Fensterrand um 20 Pixel
// verschieben
Left= 40;
for (i=0; i < 3; i++)
{
    Left+=10; // ab 50 Pixel vom linken Fensterrand verschieben
    eval('document.all.DIV_Zeiger' + i + '].style.left=' + Left);
}

```

Hinweis: Anstelle von i.toString() kann auch nur i kodiert werden, da der Browser aufgrund von document.write() i automatisch nach String umwandelt. analog für i.toString() innerhalb eval()

Die Verschiebung erfolgt natürlich mit den Objekten, welche innerhalb von <DIV> .. </DIV> kodiert wurden. Im Falle von IMG's innerhalb des DIV werden diese mit verschoben.

Vor allem **wegen** der Manipulation von HTML-Objekten werden DIV's verwendet.

Objekt style repräsentiert **nur diejenigen** zur Laufzeit **real im Element** vorhandenen **und zum Zeitpunkt aktuellen** Styles (CSS), die als Inline-Style per STYLE-Attribut oder style Objekt erzeugt wurden, aber **keinen globalen** CSS und keine Pseudoklassen.

Styles können gelesen **und** geschrieben werden

Werte von CSS-Angaben lesen: es kann nur der Wert geliefert werden, der zugewiesen wurde bzw. Standard ist

Bsp: Zuweisung von "green"
lesen von "green" und **nicht** #00FF00

erst nach dem kompletten Laden des Dokumentes

unmittelbar nach dem Setzen einer CSS-Angabe nicht möglich (Browser benötigt Zeit zur Verarbeitung):

Empfehlung: Funktionsaufruf je für Setzen und Lesen verschafft dem Browser Zeit

Hinweis: Objekt currentStyle referenziert **alle** zur Laufzeit **real im Element** vorhandenen **und zum Zeitpunkt aktuellen** Styles (CSS)

CSS kann für ein Element im Dokument definiert worden sein als

| | |
|------------------------------|---|
| im Dokument global | z.B. per userdefinierte Style-Regel im HEAD |
| im Element als Inline-Styles | per STYLE-Attribut oder style Objekt |
| Pseudoklasse | |

Script ab IE 5.x

Ob ein Objekt ein Style-Layout hat, lässt sich ab IE 5.5 anhand der Eigenschaft objekt.currentStyle.hasLayout überprüfen.

Ein Style-Layout wird erzeugt durch Kodierung einer Style-Eigenschaft

oder durch Setzen der Objekt-Eigenschaft .contentEditable auf true
(falls Eigenschaft implementiert ist zum jeweiligen Objekt).

Folgende Objekte haben immer ein Style-Layout:

| | |
|-----------------------------|---------------------------------|
| BODY, IMG, INPUT, TABLE, TD | |
| Objekte mit | .style.display auf inline-block |
| | .style.height |
| | float auf left oder right |
| | .style.position auf absolute |
| | .style.width |
| | .style.writingMode auf tb-rl |
| | .style.zoom |

Ein Objekt mit kodierten Style-Eigenschaften, also mit Style-Layout, ist mit seinen Style-Eigenschaften im Objekt currentStyle referenziert.

Die Implementation von Style-Eigenschaften ist objektspezifisch (siehe jeweilige Objektbeschreibung).

Das Objekt style hat keine Style-Eigenschaften und ist nicht in currentStyle referenziert.



Wichtige Eigenschaften des style Objektes sind .type und .media

Beispiel 1

```
document.body.style.fontFamily = "Verdana"
```

Beispiel 2

```
var FeldAllerIMGOBJekte = document.all.tags("IMG");

if (FeldAllerIMGOBJekte.length)
{
    for (var Index = 0; Index < FeldAllerIMGOBJekte.length; Index++)
    {
        var FeldElement = FeldAllerIMGOBJekte[Index];

        if (FeldElement.style.position == "absolute")
        { FeldElement.style.top = 0; }
    }
}
```

Beispiel 3

```
<DIV ID="ID_Div1" STYLE="background-color:blue;font-weight:bold">
    Div1
</DIV>
<DIV ID=" ID_Div2" STYLE="background-color:red;font-size:18pt; font-family:Verdana;">
    Div2
</DIV>
<SCRIPT>
    ID_Div1.style.cssText += (' + ID_Div1.style.cssText);
</SCRIPT>
```

Beispiel 4 für globalen Style per HEAD:

```
<HEAD>
<STYLE>
    .an {text-decoration: underline overline; color:blue;}
    .aus {text-decoration: none; color:black;}
</STYLE>
</HEAD>
<BODY>
    <A      HREF="test.htm"
           CLASS="aus"
           onmouseover="this.className='an';"
           onmouseout="this.className='aus';"
    >
    </A>
</BODY>
```

Beispiel 5 für Sound mit Sekundenanzeige:

```
<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln
    // Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    // ++++++ Routinen der Sekundenzählung
    var SekundenZahler = 0;
    var SekundenZahlerTimeoutID = null;

    function SekundenZahlen() // wird durch Rekursion alle Sekunde neu gestartet
    {
        // Zähler erhöhen
        SekundenZahler++;

        // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
        // neu berechnen und damit alle DIV's neu visualisieren
```



```

        document.recalc();
    }

function SekundenZaehlen_Start()
{
    // prüfen ob Sekunden zählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekunden zählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"

```



```

    + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
    );
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ----- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ----- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
                                        "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                        );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText", "SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                                + SoundDauerInSekunden.toString()
                                + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + ' STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

    );

    document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
                    + ' STYLE="color:hotpink;font-weight:bold"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

    );

    document.write( ' <DIV ID="ID_DIV_MessLatte"'
                    + ' STYLE="color:white;background-color:gray"'
                    + '>'
                    + '</DIV>'

    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

```




```

        // ----- Sound-Objekt wiedergeben
        SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
    }
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben //-->
</BODY>
</HTML>

```

Eine Spezialversion von frei-definierten Style ist der Customer-Tag laut Objekt custom

Folgende Style des NS unterstützt der IE **nicht**:

```

MozBinding
MozOpacity
parentRule

```

Folgende Styles des IE unterstützt der NS **nicht**:

```

accelerator
backgroundPositionXXX
BehaviorXXX
FilterXXX
imeMode
layoutFlow
layoutGridXXX
lineBreak
overflowXXX
posXXX
rubyXXX
scrollbarXXX
styleFloat
textAlignLast
textAutoSpace
textDecorationXXX
textJustify
textJustifyTrim
textKashidaXXX
textUnderlinePosition
wordBreak
wordWrap
writingMode
zoom

```

mit XXX bedeutet "alle Varianten des Style"

4.3.2.2.4.3.39.9. CSS-Konformität der Versionen Internet Explorer

Der Internet Explorer unterstützt den CSS-Standard je nach aktueller Browserversion nur teilweise.

CSS1 nur teilweise vor dem IE 6.x
 komplett ab IE 6.x (und browsereigene Erweiterungen von CSS1, die nicht in CSS1 definiert sind)

CSS2 ab IE 6.x
 bis dato nur teilweise

Hinweis: CSS1 ist in HTML4 laut W3C voll implementiert
 CSS2 ist in HTML4 bis dato noch nicht komplett implementiert worden

4.3.2.2.4.3.39.9.1. CSS1-Konformität des Internet Explorer 6.x zu seinen Vorgängern

Vorgänger des IE 6.x haben CSS1 nicht komplett implementiert
 IE 6.x hat CSS1 voll implementiert (sowie zusätzliche IE-spezifische Attribute)

Durch die Kodierung der **!DOCTYPE**-Angabe am Kopf des Dokumentes kann
 entweder die CSS1-Konformität
 oder Kompatibilität zu den Browservorgängern
 erreicht werden

siehe auch Eigenschaft .compatMode des document Objektes

IE 6.x-Kompatibilitätsmodus zu seinen Vorgängern = CSS1- Inkonformität

keine CSS1-Konformität

volle Konformität zu den Vorgängern des IE 6.x

!DOCTYPE darf **nicht** kodiert sein

Der IE 6.x verarbeitet alle **nicht** CSS1-konformen CSS-Deklarationen
 kann **keine** CSS1-konformen CSS-Deklarationen verarbeiten



IE 6.x-Inkompatibilitätsmodus zu seinen Vorgängern = CSS1- Konformität

volle CSS1-Kompatibilität

keine Kompatibilität zu den Vorgängern des IE 6.x

es **muss** !DOCTYPE kodiert sein

Der IE 6.x ignoriert alle **nicht** CSS1-konformen CSS-Deklarationen
kann **alle** CSS1-konformen CSS-Deklarationen verarbeiten

Hinweis: Vorgängerbrowser, die nicht CSS1 voll unterstützen, werden aus Sicht dieser Browser "falsch" kodierte CSS1-Deklarationen ignorieren oder als falsche Referenz bemängeln.

Empfehlung zum IE 6.x

CSS1-Elemente im IE 6.0 weglassen, die nicht in den Vorgänger verwendbar sind **und** kein !DOCTYPE kodieren.

Nur wenn ausschliesslich für den IE ab Version 6.x **und** immer CSS1-konform programmiert wird, dann !DOCTYPE kodieren.

Leider kann !DOCTYPE nicht direkt scriptgesteuert erzeugt werden (es sei denn, man erzeugt ein Dokument zur Laufzeit).

Achtung: Das Objekt document.body ist nicht mehr verfügbar und wurde durch das Objekt document.documentElement ersetzt !

document.documentElement ist der Zeiger auf den Body und nicht mehr document.body !

Beispiel zur Abfrage auf CSS1-Konformität und damit zur Verwendung von document.body bzw. document.documentElement

```
function DocTypeAbfrage()
{
    // Annahme: CSS1-Standard, also !DOCTYPE wurde kodiert
    var ZeigerAufDolument=document.documentElement;

    if (document.compatMode=="BackCompat")
    {
        // nicht CSS1-Standard, also wurde kein !DOCTYPE kodiert
        ZeigerAufDolument =document.body;
    }

    return ZeigerAufDolument;
}
```

Hinweise: document.compatMode ist nur lesbar, also !DOCTYPE nicht setzbar
Wert "CSS1Compat" ab IE 6
geliefert wenn !DOCTYPE kodiert wurde
Rendern mit CSS1 Standard (Standard-Compilant-Modus)
Wert "BackCompat" ab IE 3
Geliefert wenn kein !DOCTYPE kodiert wurde
Rendern mit Nicht-kein Standad-Compilant-Modus

4.3.2.2.4.3.39.9.2. Arten der vollen CSS1 - Konformität des Internet Explorer ab 6.x

Der Umfang der CSS1-Kompatibilität des IE 6.x kann eingeschränkt werden

CSS1- Kompatibilität außer in einem Frameset-Dokument

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

CSS1- Kompatibilität auch bei Frameset-Dokument

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN">
```

CSS1- Konformität genau im Umfang von HTML 4 laut World Wide Web Consortium (W3C) (auch bei Frameset-Dokument)

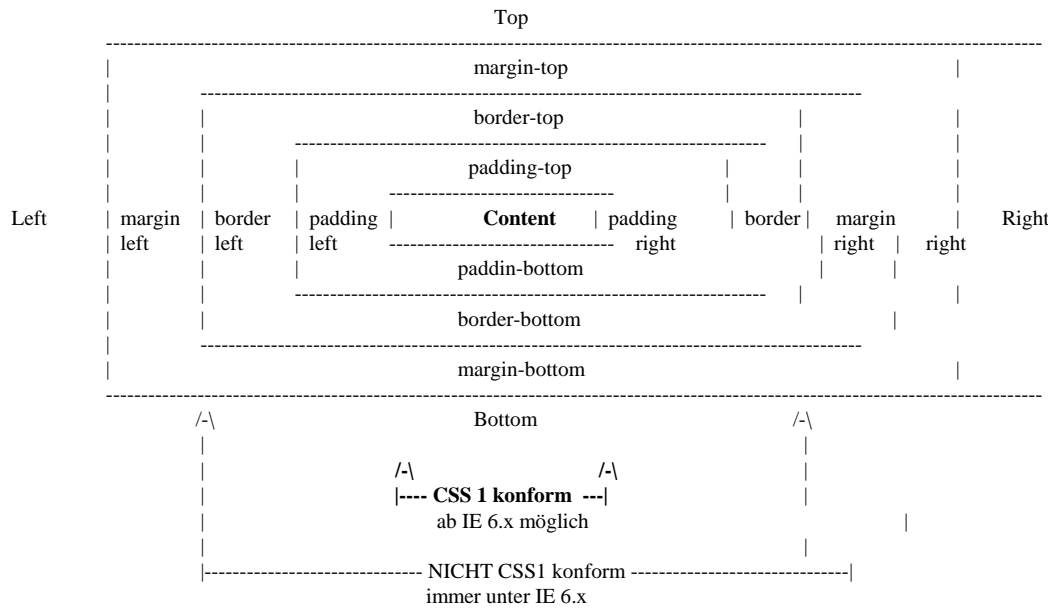
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Strict//EN">
```

oder <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">

Strict ist optional



4.3.2.2.4.3.39.9.3. CSS und Attribute width und hight

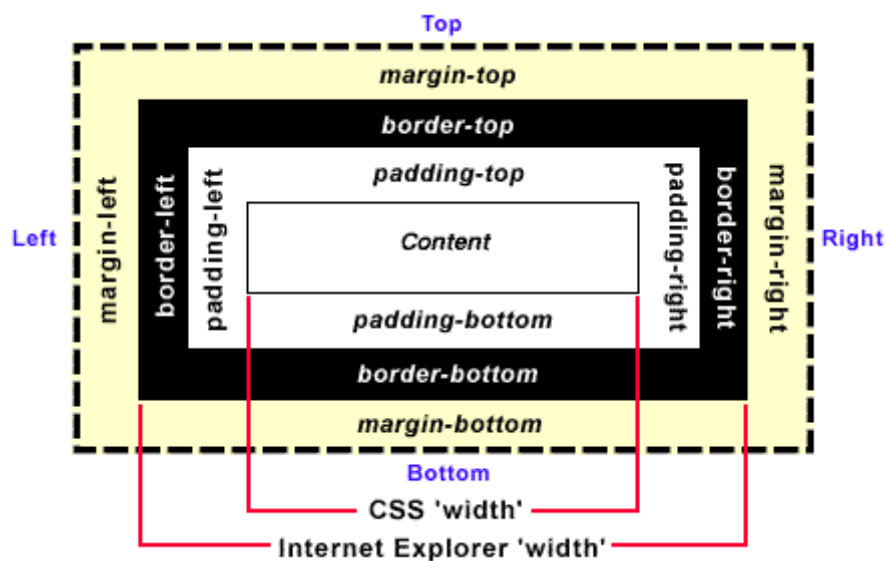
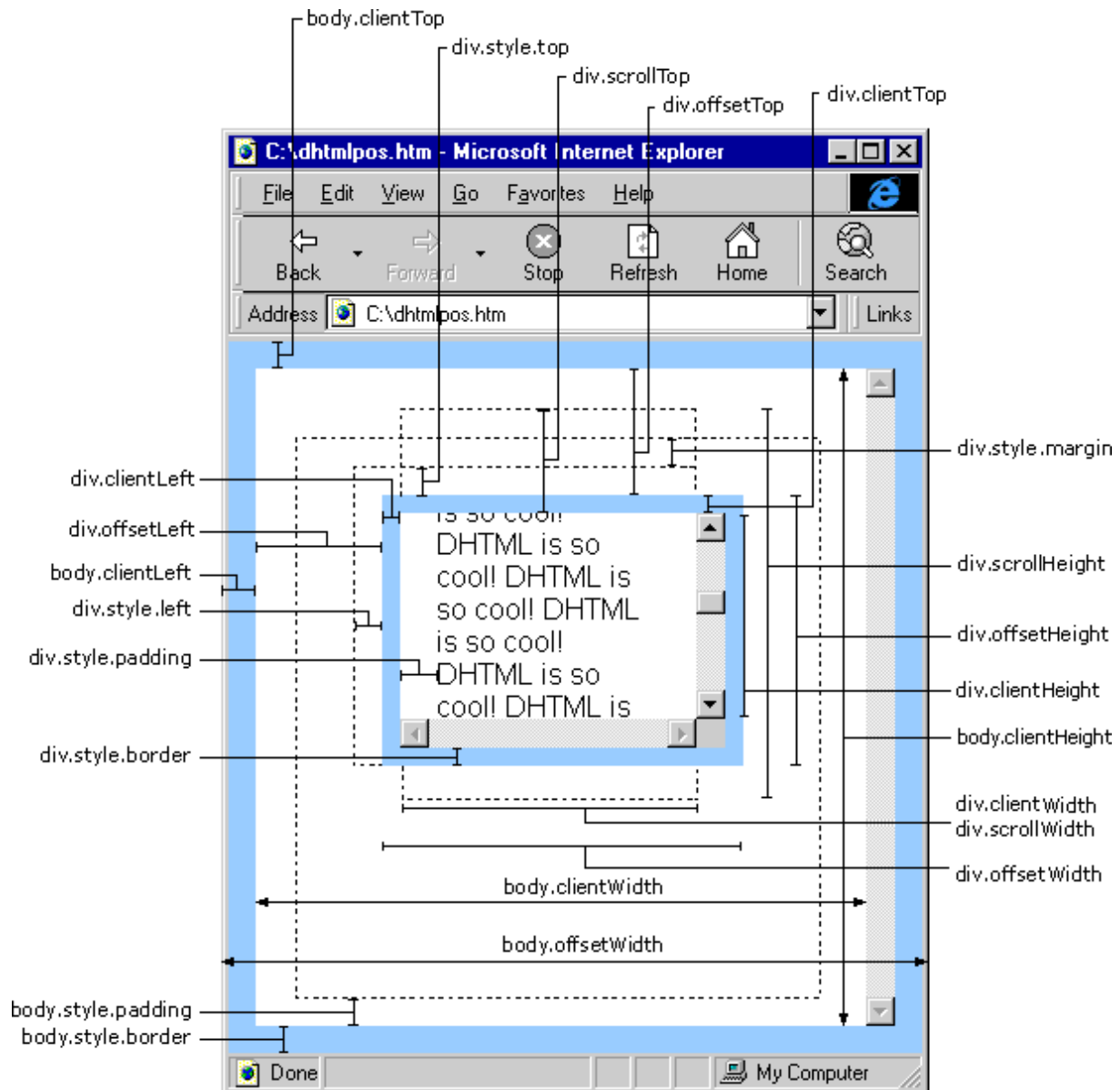


vor dem IE 6.0: nur **nicht**-CSS1-konform
width und hight : inklusive ab border

ab dem IE 6.0: CSS1-konform:
width und height: nur der **Content**

Hinweis Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objekthinhaltes zum Aussenrand





4.3.2.2.4.3.39.10. Kommentare innerhalb <STYLE> <STYLE>



```
/* Das ist ein
   CSS1-konformer
   Kommentar */
```

```
<!--
    Diese Form des Kommentares NUR wählen wenn Code ausgeblendet werden
    soll für ältere Browser,
    da nicht CSS1-konformer Kommentart
-->
```

4.3.2.2.4.3.39.11. Kodierung von Werten mit erlaubter Einheit

(Leerzeichen zwischen Einheit und Wert)

vor IE 6.x Leerzeichen darf kodiert werden
 Bsp.: "11 px" wird erkannt da nicht CSS1-konform
 "11px" wird immer erkannt

ab IE 6.x Leerzeichen darf **nur** kodiert werden, wenn **nicht** !DOCTYPE kodiert wurde
 Bsp.: "11 px" wird **nur** erkannt wenn **nicht** !DOCTYPE kodiert wurde,
 da nicht CSS1-konform
 "11px" wird immer erkannt

Empfehlung: generell keine Leerzeichen kodieren

4.3.2.2.4.3.39.12. Kodierung von CSS-Werten in Style-Sheets (Klammerung in " " bzw. ' ')

vor IE 6.x CSS1-nicht-konform (" " bzw. ' ' sind **Pflicht**kodierung):

Bsp.: DIV.George { color: "red"; font-variant: "small-caps"; }

ab IE 6.x CSS1-konform **nur** wenn !DOCTYPE kodiert wurde im Kopf des Dokumentes
 (" " bzw. ' ' dürfen **nicht** kodiert werden, da sonst
 vom Browser ignoriert !)

Bsp.: DIV.George { color: red; font-variant: small-caps; }

CSS1-nicht-konform **nur** wenn **nicht** !DOCTYPE kodiert wurde im Kopf des Dokumentes
 (" " bzw. ' ' sind **Pflicht**kodierung):

Bsp.: DIV.George { color: "red"; font-variant: "small-caps"; }

4.3.2.2.4.3.39.13. RGB-Farbangaben

vor IE 6.x Es ist möglich das im Attribut das führende "#" weggelassen wird, es sei denn
 die Attribut-Beschreibung besteht auf führendem "#".

ab IE 6.x CSS1-nicht-konform **nur** wenn **nicht** !DOCTYPE kodiert wurde im Kopf des Dokumentes:
 Es ist möglich das im Attribut das führende "#" weggelassen wird, es sei denn
 die Attribut-Beschreibung besteht auf führendem "#".

CSS1-konform **nur** wenn !DOCTYPE kodiert wurde im Kopf des Dokumentes:
 immer mit führendem "#"

4.3.2.2.4.3.39.14. CLASS-Attribut und ID-Attribut-Wert

vor IE 6.x Wert darf kann mit Ziffer beginnen
 für Wert ist Gross- und Kleinschreibung egal

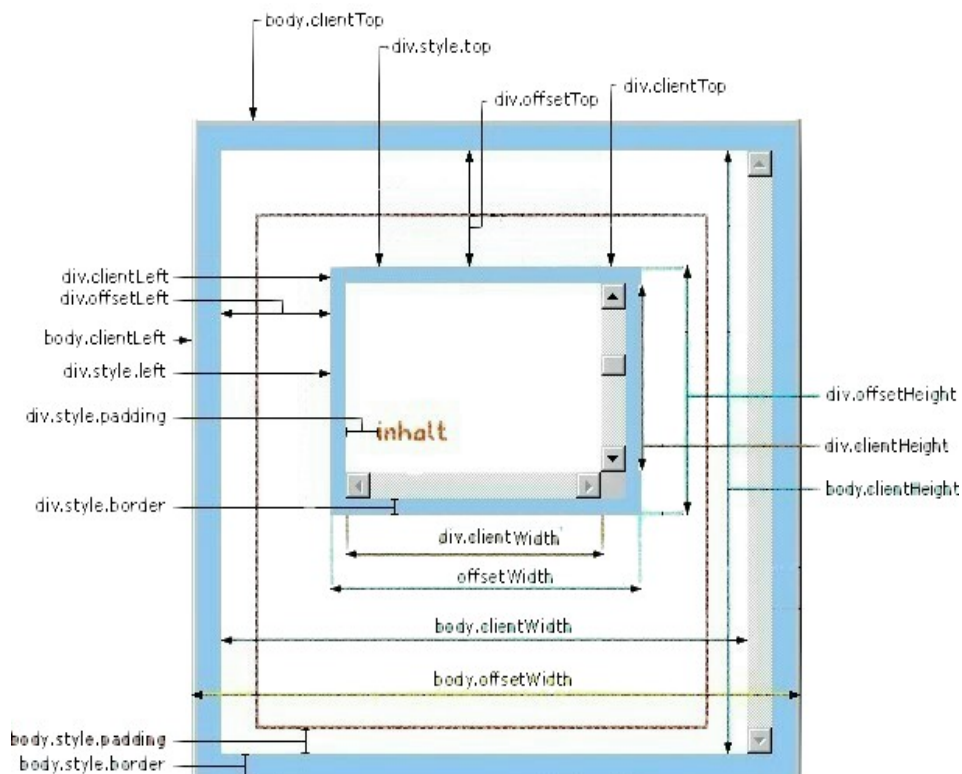
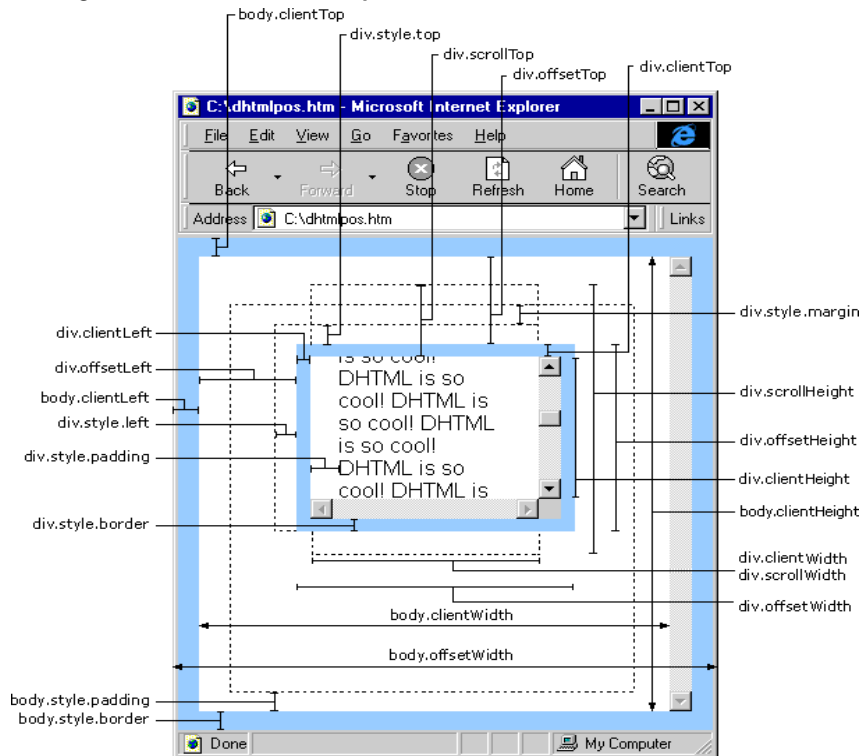
ab IE 6.x Wert darf nicht mit Ziffer beginnen
 für Wert ist Gross- und Kleinschreibung zu beachten

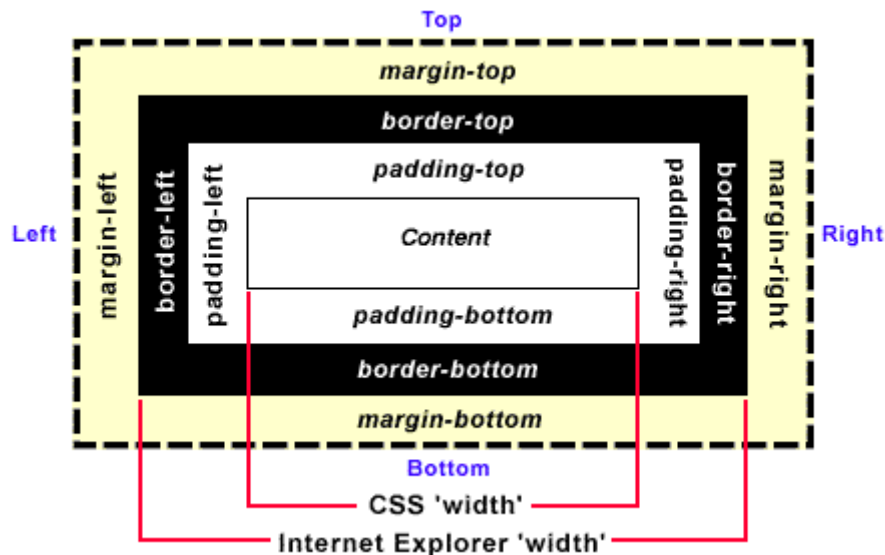
4.3.2.2.4.3.39.15. Verwendung von \

CSS1-nicht-konform	z.B. \"	Entwertung
CSS1-konform	z.B. \\0009	Unicode für TAB



4.3.2.2.4.3.39.16. Die Eigenschaften des Internet Explorer als Grafiken





4.3.2.2.4.3.39.17. Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !!!

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

4.3.2.2.4.3.39.18. Dynamischen Eigenschaftenveränderung zur Laufzeit

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

4.3.2.2.4.3.39.19. Eigenschaften

In nachfolgenden Beschreibungen wurde im HTML-Syntax die Kodierung von STYLE= aus Übersichtlichkeit weggelassen.

In HTML sind weniger Style-Eigenschaften per STYLE-Attribut kodierbar als per Script-Anweisung.

Es gibt Style-Eigenschaften, die nur in HTML kodierbar sind.

Im STYLE-Attribut kodierte Style-Eigenschaft muss hinter dem Bezeichner immer ein Doppelpunkt kodiert haben, der die Liste der Werte zur Style-Eigenschaft vom Bezeichner der Style-Eigenschaft trennt.

Bsp.: `<DIV STYLE="background-color: red;font-size:18pt; font-family:Verdana;">`
 Testtext
`</DIV>`

In HTML müssen Style-Klassen im HEAD kodiert sein, wobei die Klasse im Tag benutzt wird.

Beispiel zur Style-Kodierung:

Beispiel 1:

```
<HEAD>
<STYLE>
    .style1 { background:beige url(test.jpg) no-repeat top center}
    .style2 { background:ivory url(test.jpeg) no-repeat bottom right}
</STYLE>
</HEAD>
<BODY>
<SPAN onmouseover="this.className='style1'"
onmouseout="this.className='style2'"
>
    Testtext
</SPAN>
```

Beispiel 2:

```
<SPAN onclick="this.style.background='beige url(test.jpeg) no-repeat top center'">
```



Testtext

Für die Kodierung der Werte der Eigenschaften wird in den nachfolgenden Beschreibungen zur Vereinfachung immer ein Stringwert in " " bzw. ' ' verwendet.

Für die exakte Kodierung aber bitte **unbedingt** beachten:

Leerzeichen zwischen Einheit und Wert in der Kodierung von Werten mit erlaubter Einheit:

vor IE 6.x Leerzeichen darf kodiert werden
Bsp.: "11 px" wird erkannt da nicht CSS1-konform
 "11px" wird immer erkannt

ab IE 6.x Leerzeichen darf **nur** kodiert werden, wenn **nicht** !DOCTYPE kodiert wurde
Bsp.: "11 px" wird **nur** erkannt wenn **nicht** !DOCTYPE kodiert wurde,
 da nicht CSS1-konform
 "11px" wird immer erkannt

Empfehlung: generell keine Leerzeichen kodieren

Klammerung in " " bzw. ' ' in der Kodierung von CSS-Werten:

vor IE 6.x CSS1-nicht-konform (" " bzw. ' ' sind **Pflicht**kodierung):
Bsp.: DIV.George { color: "red"; font-variant: "small-caps"; }

ab IE 6.x CSS1-konform **nur** wenn !DOCTYPE kodiert wurde im Kopf des Dokumentes
 (" " bzw. ' ' dürfen **nicht** kodiert werden, da sonst vom Browser ignoriert !)
Bsp.: DIV.George { color: red; font-variant: small-caps; }

CSS1-nicht-konform **nur** wenn **nicht** !DOCTYPE kodiert wurde im Kopf des Dokumentes
 (" " bzw. ' ' sind **Pflicht**kodierung):
Bsp.: DIV.George { color: "red"; font-variant: "small-caps"; }

RGB-Farbangaben:

vor IE 6.x Es ist möglich das im Attribut das führende "#" weggelassen wird, es sei denn die Attribut-Beschreibung besteht auf führendem "#".

ab IE 6.x CSS1-nicht-konform **nur** wenn **nicht** !DOCTYPE kodiert wurde im Kopf des Dokumentes:
 Es ist möglich das im Attribut das führende "#" weggelassen wird, es sei denn die Attribut-Beschreibung besteht auf führendem "#".

CSS1-konform **nur** wenn !DOCTYPE kodiert wurde im Kopf des Dokumentes:
immer mit führendem "#"

4.3.2.2.4.3.39.19.1. Eigenschaften komplett

Der in nachfolgenden Beschreibungen wegen Vereinfachung verwendete Zeiger `object.style.eigenschaft` setzt voraus, dass für `object` eine konkrete Referenz kodiert werden muss.

Hinweis zu Scrollbalken: Neu erscheinende Scrollbalken verkleinern die Dimension des Browserfensters. Das ist zu beachten, wenn die Fensterdimension ohne bereits existierende Scrollbalken ermittelt wurde.

Hinweis zum Lesen einer Style-Eigenschaft:

Falls eine Style-Eigenschaft lesbar ist, so kann der Wert der Style-Eigenschaft ermittelt werden. Allerdings besteht die Möglichkeit, dass bei einem numerischen Wert, der sich auf eine Einheit z.B. px bezieht, entgegen aller Erwartungen dieser nicht numerisch, sondern als String aus numerischem Wert **und** der Werteinheit geliefert wird (z.B. beim Lesen von `.style.top`). Soll dieser Stringwert wieder zugewiesen werden, dann scheitert die Zuweisung, wenn ein numerischer Wert erwartet wird. Typische Browsermeldung dazu ist der Hinweis auf ein ungültiges Argument. Lesen und Schreiben einer Style-Eigenschaft können sich also prinzipiell unterscheiden.

Um das Problem generell zu umgehen, empfiehlt es sich, den Wert der Style-Eigenschaft ab Erstbelegung im gesamten Verlauf der Veränderung der Style-Eigenschaft in eine numerische Variable zwischenspeichern und diese bei Bedarf auszulesen sowie bei **jeder** Wertzuweisung zur Style-Eigenschaft zu verwenden. Desweiteren wird empfohlen, keine **direkten** numerischen Operationen mit Style-Eigenschaften durchzuführen (falls die Style-Eigenschaft keinen String, sondern einen numerischen Wert liefert). Die Zwischenspeicherung in eine numerische Variable, mit der operiert und das Ergebnis auf die Style-Eigenschaft zugewiesen wird, ist die bessere Lösung und umgeht **immer** o.g. Problem.



Man beachte die Kodierungsvorschriften bei Erstbelegung der Style-Eigenschaft bezüglich anzugebender Einheit z.B. 'px'.

:active	Pseudoklasse des a Objektes für aktiven Link (User hat Link aktiviert) ist nicht per Script ansprechbar
:first-letter	Pseudoklasse für Style des ersten Buchstaben eines HTML-Elementes (Tags) nicht in Script ansprechbar nur für Blockelemente wie DIV und SPAN Elemente mit Eigenschaft .style.display mit Wert "block" ab IE 5.5
:first-line	Pseudoklasse für Style der ersten Zeile eines HTML-Elementes (Tags) nicht in Script ansprechbar nur für Blockelemente wie DIV und SPAN Elemente mit Eigenschaft .style.display mit Wert "block" ab IE 5.5
:hover	Pseudoklasse des a Objektes für Link, der nicht aktiviert wurde solange Mauszeiger auf dem Link steht, ist die Klasse aktiv ist nicht per Script ansprechbar
:link	Pseudoklasse des a Objektes für Link, der nicht kürzlich aktiviert wurde. ist nicht per Script ansprechbar ab IE 4.x
:visited	Pseudoklasse des a Objektes für Link, der bereits aktiviert wurde und User ist wieder auf das Dokument zurückgekehrt, das den Link enthält. ist nicht per Script ansprechbar
.media	Media-Typ für Ausgabe eines Objektes
.style.accelerator	ALT+Taste als Kurztastenkombination-Zugriff auf Objekt ein/aus
.style.background	Hintergrund eines Objektes bei P, DIV, SPAN: Eigenschaft nur für eingeschlossenen Text
.style.backgroundAttachment	Hintergrundbild nur ab IE 4.x
.style.backgroundColor	Hintergrundfarbe nur ab IE 4.x Achtung: Für das body Objekt gilt: BGCOLOR Attribut und Eigenschaft .bgColor für die Hintergrundfarbe im BODY sind deprecated und durch <BODY STYLE="background-color:....."> zu ersetzen.
.style.backgroundImage	Hintergrundbild, das sich über die Hintergrundfarbe legt nur ab IE 4.x
.style.backgroundPosition	Hintergrund-Position für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundPositionX	Hintergrund-Position X für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundPositionY	Hintergrund-Position Y für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundRepeat	Kachelung des Hintergrundbildes
.style.behavior	Ort der Behaviors-Datei *.htc (Verhaltensweise eines Elementes) bzw. Standard-Behavior des IE
.style.border	Rahmen alle 4 Seiten: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottom	Rahmen unten: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomColor	Rahmen unten: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomStyle	Rahmen unten: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomWidth	Rahmen unten: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderCollapse	Rahmen alle 4 Seiten: Verschmelzung zum Single Border
.style.borderColor	Rahmen alle 4 Seiten: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeft	Rahmen links: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftColor	Rahmen links: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftStyle	Rahmen links: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftWidth	Rahmen links: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des



	Objektes besitzen
.style.borderRight	Rahmen rechts: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightColor	Rahmen rechts: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightStyle	Rahmen rechts: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightWidth	Rahmen rechts: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderStyle	Rahmen alle 4 Seiten: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTop	Rahmen oben: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopColor	Rahmen oben: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopStyle	Rahmen oben: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopWidth	Rahmen oben: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderWidth	Rahmen alle 4 Seiten: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.bottom	Abstand unterer Objektrand zur Umgebung Oberkante Objekt muss position-Eigenschaft kodiert haben identisch mit Attribut .style.pixelBottom und .style.posBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.clear	Textumfluss links, rechts siehe auch Eigenschaften .style.whiteSpace .style.float .style.styleFloat
.style.clip	Position sichtbarer Bereich über dem Objekt Position immer bezüglich Umgebung Objekt muss absolut positioniert sein ! ausserhalb des Clipfensters wird alles transparent siehe auch Eigenschaften .style.overflow .style.overflowX .style.overflowY .style.display .style.visibility
.style.color	Vordergrundfarbe (Textfarbe)
.style.cssText	Wert des STYLE-Attributes im Tag des HTML-Elementes
.style.cursor	Mauscursor Arten z.T. erst ab IE 6.x
.style.direction	Horizontale Richtung des Renderns z.B. vom Plain-Textteil eines Elementes, also Leserichtung für den User einstellen (z.B. von rechts nach links) siehe auch Eigenschaft .style.unicodeBidi
.style.display	Art der Anzeige (des Renderns) eines Objektes (auch mit Ausrichtung) siehe auch Eigenschaften .style.overflow .style.overflowX .style.overflowY .style.clip .style.visibility z.T. ab IE 6.x Hinweis: Wenn !DOCTYPE am Dokumentanfang kodiert wurde, so darf .style.display nicht auf "list-item" gesetzt sein. Wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde, so kann .style.display auf "list-item" gesetzt sein !DOCTYPE am Dokumenten-Anfang ist erst ab IE 6.x kodierbar !
.style.filter	Filter des Internet Explorer implementieren (siehe Objekt filter)
float	Textumfluss links, rechts unter IE 5.x : DIV und SPAN müssen Eigenschaft .style.width besitzen



	siehe auch Eigenschaften	.style.whiteSpace .style.clear .style.styleFloat
.style.font	Textfont	komplett mit allen Teileigenschaften (ebenfalls Style-Eigenschaften) oder Standard-Textfont
.style.fontFamily	Textfont	Familie
.style.fontSize	Textfont	Höhe
.style.fontStyle	Textfont	Stil
.style.fontVariant	Textfont	Stil-Variante
.style.fontWeight	Textfont	Fettheit
.style.height	Höhe Objekt	identisch mit .style.pixelHeight und .style.scrollHeight, aber .style.height: String .style.pixelHeight: Integer, nur Pixel .style.scrollHeight: Float pointing und Integer
	Hinweis:	Wenn !DOCTYPE am Dokumentanfang kodiert wurde, so kann .style.height auf "auto" gesetzt sein Wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde, so darf .style.height nicht auf "auto" gesetzt sein !DOCTYPE am Dokumenten-Anfang ist erst ab IE 6.x kodierbar !
.style.imeMode	Status des Input Method Editor (IME) für Eingabe von Chinese, Japanese oder Korean Zeichen	
.style.layoutFlow	deprecated	zu ersetzen durch Eigenschaften .style.writingMode Ost-Asiatische Darstellung (Flussrichtung) siehe auch Eigenschaft .style.verticalAlign für Ausrichtung vertikal für Objekte mit VALIGN-Attribut
.style.layoutGrid	Textzeichen-Layout-Gitter z.B. für asiatische Sprachen	
.style.layoutGridChar	Textzeichen-Layout-Gitter	Zeichengröße nur für Blockelemente wie SPAN und DIV etc. verlangt .style.layoutGridMode auf Wert "line" oder "both"
.style.layoutGridLine	Textzeichen-Layout-Gitter	Linie nur für Blockelemente wie SPAN und DIV etc. verlangt .style.layoutGridMode auf Wert "line" oder "both"
.style.layoutGridMode	Textzeichen-Layout-Gitter	2D
.style.layoutGridType	Textzeichen-Layout-Gitter	Typ nur für Block-Elemente wie SPAN, DIV
.style.left	Abstand linker Objektrand zur Umgebung rechte Kante	Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes, wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal) Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV): Abstand der linken Kante des IMG zur linken Kante des DIV Objekt muss Eigenschaft .style.position kodiert haben identisch zu .style.pixelLeft und .style.posLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer
.style.letterSpacing	Textzeichen	Abstand
.style.lineBreak	Zeilenumbruch	
.style.lineHeight	Abstand zweier Objekte oder zweier Textzeilen	ab IE 4.x
.style.listStyle	Listendarstellung	Aufzählungsliste-Elemente benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStyleImage	Listendarstellung	Marker als Image siehe auch .style.listStyleType benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStylePosition	Listendarstellung	Marker Position benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStyleType	Listendarstellung	Markertyp (nicht Image) siehe auch .style.listStyleTypeImage benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.margin	Aussenrand	Abstand des Aussenrandes links, rechts, oben, unten des Objektes von anderen Objekten im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.marginBottom	Aussenrand	Abstand unten zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Margin: Abstand des Aussenrandes eines Objektes zur Umgebung



.style.marginLeft	<p>Padding: Abstand des Objektinhaltes zum Aussenrand Aussenrand Abstand links zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen</p> <p>Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>
.style.marginRight	<p>Padding: Abstand des Objektinhaltes zum Aussenrand Aussenrand Abstand rechts zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen</p> <p>Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>
.style.marginTop	<p>Padding: Abstand des Objektinhaltes zum Aussenrand Aussenrand Abstand oben zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen</p> <p>Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>
.style.minHeight	<p>Padding: Abstand des Objektinhaltes zum Aussenrand minimale Höhe des Objektes ab IE 6.x und nur wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde und dann auch nur für TD, TH und TR innerhalb eines fixierten Tabellenlayouts Tabelle mit fixiertem Layout erzeugen per Eigenschaft .style.tableLayout mit Wert "fixed" wird schneller dargestellt als bei Auto-Layout Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto" also Auto-Layout, also kein fixiertes Tabellenlayout</p>
.style.onOffBehavior	<p>deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound</p>
.style.overflow	<p>Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster. Anzeige des Objektes mit/ohne Scrollelemente vertikal und horizontal Anzeige des Objektes ein/aus siehe auch Eigenschaften .style.overflowX .style.overflowY .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.overflowX	<p>ab IE 5.x Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster. Anzeige des Objektes mit/ohne Scrollelemente nur horizontal Anzeige des Objektes ein/aus siehe auch Eigenschaften .style.overflow .style.overflowY .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.overflowY	<p>ab IE 5.x Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster. Anzeige des Objektes mit/ohne Scrollelemente nur vertikal Anzeige des Objektes ein/aus siehe auch Eigenschaften .style.overflow .style.overflowX .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.padding	<p>ab IE 5.x Abstand links, rechts, oben, unten zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für img Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>
.style.paddingBottom	<p>Padding: Abstand des Objektinhaltes zum Aussenrand Abstand unten zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für img Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objektinhaltes zum Aussenrand</p>



.style.paddingLeft	Abstand links zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für img Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.paddingRight	Abstand rechts zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für img Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.paddingTop	Abstand oben zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für img Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.pageBreakAfter	Seitenumbruch nach dem Druck des Objektes beim Druck des Dokumentes nicht möglich für BR und HR-Objekt siehe auch Eigenschaften .style.whiteSpace .style.wordBreak .style.wordWrap .style.pageBreakBefore
.style.pageBreakBefore	Seitenumbruch vor dem Druck des Objektes beim Druck des Dokumentes nicht möglich für BR und HR-Objekt siehe auch Eigenschaften .style.whiteSpace .style.wordBreak .style.wordWrap .style.pageBreakAfter
.style.pixelBottom	Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und style.posBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.pixelHeight	Objekthöhe identisch mit .style.heigth und .style.posHeigt aber .style.heigt: String .style.pixelHeigt: Integer, nur Pixel .style.posHeigth: Float pointing und Integer
.style.pixelLeft	Abstand linker Objektrand zur Umgebung rechte Kante Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes, wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal) Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV): Abstand der linken Kante des IMG zur linken Kante des DIV identisch mit Attribut .style.left und .style.posLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer
.style.pixelRight	Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut .style.rigth und .style.posRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer
.style.pixelTop	Abstand oberer Objektrand zur Umgebung Unterkante Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes, wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal) Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV): Abstand der oberen Kante des IMG zur oberen Kante des DIV identisch mit Attribut .style.top und .styleposTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer
.style.pixelWidth	Objektbreite identisch mit .style.width und .style.posWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer
.style.posBottom	Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und .style.pixelBottom aber .style.bottom: String .stylepixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.posHeight	Objekthöhe identisch mit .style.heigth und .style.pixelHeigt aber .style.heigt: String .style.pixelHeigt: Integer, nur Pixel



	.style.posHeigh:	Float pointing und Integer
.style.position	Art der Objektpositionierung innerhalb Eltern z.B. BODY	
.style.posLeft	Abstand linker Objektrand zur Umgebung rechte Kante	
	Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes,	
	wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal)	
	Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):	
	Abstand der linken Kante des IMG zur linken Kante des DIV	
	identisch mit Attribut .style.left und .style.pixelLeft	
	aber .style.left:	String
	.style.pixelLeft:	Integer, nur Pixel
	.style.posLeft:	Float pointing und Integer
.style.posRight	Abstand rechter Objektrand zur Umgebung linke Kante	
	identisch mit Attribut .style.rigth und .style.pixelRight	
	aber .style.right:	String
	.style.pixelRight:	Integer, nur Pixel
	.style.posRight:	Float pointing und Integer
.style.posTop	Abstand oberer Objektrand zur Umgebung Unterkante	
	Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes,	
	wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal)	
	Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):	
	Abstand der oberen Kante des IMG zur oberen Kante des DIV	
	identisch mit Attribut .style.top und .style.pixelTop	
	aber .style.top:	String
	.style.pixelTop:	Integer, nur Pixel
	.style.posTop:	Float pointing und Integer
.style.posWidth	Objektbreite	
	identisch mit .style.width und .style.pixelWidth	
	aber .style.width:	String
	.style.pixelWidth:	Integer, nur Pixel
	.style.posWidth:	Float pointing und Integer
.style.right	Abstand rechter Objektrand zur Umgebung linke Kante	
	benötigt kodiertes .style.position Attribut	
	identisch mit Attribut .style.posRigth und .style.pixelRight	
	aber .style.right:	String
	.style.pixelRight:	Integer, nur Pixel
	.style.posRight:	Float pointing und Integer
.style.rubyAlign	Ausrichtung des ruby Objektes	
	hier nicht weiter erklärt	
	im STYLE-Attribut ruby-align kodieren	
.style.rubyOverhang	Überhang des ruby Objektes	
	hier nicht weiter erklärt	
	im STYLE-Attribut ruby-overhang kodieren	
.style.rubyPosition	Position des ruby Objektes	
	hier nicht weiter erklärt	
	im STYLE-Attribut ruby-position kodieren	
.style.scrollbar3dLightColor	xxxxxxxxxx	x scrollbar3dLightColor
	yyyyyyyyyyw	y scrollbarHightlightColor
	xy.....zw scrollbarFaceColor
	xy...../ \.....zw	/ \ scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw	xy.....zw
	xzzzzzzzzzw	z scrollbarShadowColor
	wwwwwwwww	w scrollbarDarkShadowColor
.style.scrollbarArrowColor	xxxxxxxxxx	x scrollbar3dLightColor
	yyyyyyyyyyw	y scrollbarHightlightColor
	xy.....zw scrollbarFaceColor
	xy...../ \.....zw	/ \ scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw	xy.....zw
	xzzzzzzzzzw	z scrollbarShadowColor
	wwwwwwwww	w scrollbarDarkShadowColor
.style.scrollbarBaseColor	Basisfarbe aller Scrollbalken-Elemente	
	automatische Farbstafflung für sämtliche Teilkomponentender Scrollbalken-Elemente	
.style.scrollbarDarkShadowColor	xxxxxxxxxx	x scrollbar3dLightColor
	yyyyyyyyyyw	y scrollbarHightlightColor
	xy.....zw scrollbarFaceColor
	xy...../ \.....zw	/ \ scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw	xy.....zw
	xzzzzzzzzzw	z scrollbarShadowColor
	wwwwwwwww	w scrollbarDarkShadowColor
.style.scrollbarFaceColor	xxxxxxxxxx	x scrollbar3dLightColor
	yyyyyyyyyyw	y scrollbarHightlightColor
	xy.....zw scrollbarFaceColor
	xy...../ \.....zw	/ \ scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw	xy.....zw



	xzzzzzzzzw	z	scrollbarShadowColor
	wwwwwwwww	w	scrollbarDarkShadowColor
.style.scrollbarHighlightColor	xxxxxxxxxxx	x	scrollbar3dLightColor
	yyyyyyyyyyw	y	scrollbarHightlightColor
	xy.....zw	scrollbarFaceColor
	xy...../\.....zw	/\	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw		
	xzzzzzzzzw	z	scrollbarShadowColor
	wwwwwwwww	w	scrollbarDarkShadowColor
.style.scrollbarShadowColor	xxxxxxxxxxx	x	scrollbar3dLightColor
	yyyyyyyyyyw	y	scrollbarHightlightColor
	xy.....zw	scrollbarFaceColor
	xy...../\.....zw	/\	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
	xy.....zw		
	xzzzzzzzzw	z	scrollbarShadowColor
	wwwwwwwww	w	scrollbarDarkShadowColor
.style.scrollbarTrackColor	Farbe des Trackelementes von Scrollbalken (Track = ziehen)		
.style.styleFloat	Textumfluss links, rechts		
.style.tableLayout	Tabellen-Layout (auto oder fixed)		
	siehe auch Eigenschaft .style.minHeight		
	Tabelle mit fixiertem Layout erzeugen per Eigenschaft .style.tableLayout mit Wert "fixed"		
	wird schneller dargestellt als bei Auto-Layout		
	Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto"		
	also Auto-Layout, also kein fixiertes Tabellenlayout		
.style.textAlign	Textausrichtung nur für Block-Elemente wie SPAN oder DIV		
	siehe auch .style.textAlignLast und .style.textJustify		
.style.textAlignLast	Textausrichtung der letzten Zeile		
	nur für Block-Elemente wie SPAN oder DIV		
	siehe auch .style.textAlign und .style.textJustify		
.style.textAutospace	Zusatz-Textabstand bei asiatischen Zeichen		
.style.textDecoration	dekoratives Layout eines Textes für nichtleere Textobjekte (unterstreichen, überstreichen, durchstreichen)		
	z.B. ist ein leeres		
	bei Block-Element: Layout wird an Kinder vererbt		
	siehe auch Eigenschaft .style.textDecorationBlink		
.style.textDecorationBlink	Wert der Eigenschaft .style.textDecoration auf "blink" prüfen		
	Text auf blinkend prüfen		
.style.textDecorationLineThrough	Textdekoration "line-through" (durchstreichen)		
.style.textDecorationNone	Textdekoration "none" (deaktivieren aller aktiven Textdekorationen)		
.style.textDecorationOverline	Textdekoration "overline" (überstreichen)		
.style.textDecorationUnderline	Textdekoration "underline" (unterstreichen)		
	siehe auch Eigenschaft .style.textUnderlinePosition		
.style.textIndent	Position-Ident in der der ERSTEN Zeile eines Textes		
	Ident nicht möglich in der Mitte eines umgebrochenen Objekts		
	(Umbrechung z.B. per BR)		
.style.textJustify	Textausrichtung Blocksatz nur für Blockelemente wie SPAN oder DIV		
	Attribut .style.textAlign muss Wert "justify" haben		
.textKashidaSpace	Typographischer Effekt "Kashida" für Arabisch		
	siehe auch Eigenschaft .style.textJustify		
	nur für Blockelemente wie DIV und SPAN		
.style.textOverflow	Rahmen um Text		
	ab IE 6.x		
	Eigenschaft ermöglicht einen Rahmen um einen Text, wobei der Text im Rahmen		
	überlaufen kann		
	Überlauf und Rahmen können folgende Formen haben:		
	Abschneiden an Rahmengrenze		
	Abschneiden an Rahmengrenze mit automatischem Anfügen von "..."		
	als Andeutung einer Fortsetzung		
	kein Abschneiden, also Rahmen in voller Textbreite		
	Achtung: Damit der Text nicht umgebrochne wird im Rahmen, muss die Eigenschaft		
	.style.whiteSpace auf "nowrap" gesetzt sein		
	bzw. der Text in das NOBR-Tag eingeschlossen sein.		
	Desweiteren sollte die Eigenschaft .style.overflow auf "hidden" gesetzt sein,		
	damit keine Scrollbalken erscheinen (im Gegensatz zu "auto" oder		
	"scroll")		
.style.textTransform	Textkonvertierung nach Gross oder Klein oder erstes Zeichen im Wort stets groß		
	siehe auch Objekt String		
.style.textUnderlinePosition	Position der Textdekoration "underline" (unterstreichen)		
	siehe auch Eigenschaft .style.textDecorationUnderline		
.style.top	Abstand oberer Objektrand zur Umgebung Unterkante		
	Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes,		
	wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal)		
	Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):		
	Abstand der oberen Kante des IMG zur oberen Kante des DIV		
	benötigt kodierte Eigenschaft .style.position		



	identisch mit Attribut <code>.style.posTop</code> und <code>.style.pixelTop</code>
	aber <code>.style.top</code> : String
	<code>.style.pixelTop</code> : Integer, nur Pixel
	<code>.style.posTop</code> : Float pointing und Integer
<code>.style.unicodeBidi</code>	Bidirektionale Dateneingabe von Unicode in das Element (Zeichenstrom, der vom Elternobjekt in das Element fließt, kann seine eigene Flussrichtung besitzen).
	Typische Anwendung ist z.B. der Mix aus Zeichen, die mal von links nach rechts und mal von rechts nach links gerendert werden sollen. Damit ist für den User eine variable Leserichtung möglich, falls der User beide Richtungen benutzen will.
	Es ist möglich, den Datenstrom von Zeichen trotz aktiver bidirektionaler Dateneingabe genau nach Vorgabe laut Eigenschaft <code>.style.direction</code> fließen zu lassen. Bsp.: Wenn <code>.style.direction</code> auf "ltr" gesetzt ist, so wird diese Richtung konsequent eingehalten, auch wenn der Datenstrom von rechts nach links fließen will.
	Die Eigenschaft <code>.style.direction</code> sollte auf "inherit" gesetzt sein, wenn bidirektionaler Datenfluss immer erfolgreich sein soll und die Flussrichtung des Datenstromes unbekannt ist. Damit wird die Richtung des Elternobjektes übernommen.
<code>.style.verticalAlign</code>	Ausrichtung vertikal für Objekte
	mit VALIGN-Attribut
	und bei Textobjekten ohne VALIGN-Attribut
<code>.style.visibility</code>	Objekt-Sichtbarkeit ohne Reservierung des Platzs im Layout der Umgebung ab IE 5.x Sichtbarkeit eines Kindobjektes, auch wenn Eltern unsichtbar sind siehe auch Eigenschaften <code>.style.overflow</code> <code>.style.overflowX</code> <code>.style.overflowY</code> <code>.style.clip</code> <code>.style.display</code> Hinweis: <code>display:none</code> Objekt behält den Platz im Layout der Umgebung, obwohl Objekt unsichtbar ist
<code>.style.whiteSpace</code>	automatischer Zeilenumbruch bei Block-Elementen wie DIV und SPAN Umbrechen erfolgt an den Stellen wo Blanks, Tabs stehen, aber nicht an der Stelle eines geschützten Blanks () siehe auch Eigenschaften <code>.style.pageBreakAfter</code> <code>.style.pageBreakBefore</code> <code>.style.wordBreak</code> <code>.style.wordWrap</code>
<code>.style.width</code>	Objektbreite identisch mit Attribut <code>.style.posWidth</code> und <code>.style.pixelWidth</code> aber <code>.style.width</code> : String <code>.style.pixelWidth</code> : Integer, nur Pixel <code>.style.posWidth</code> : Float pointing und Integer
<code>.style.wordBreak</code>	Zeilenumbruch in Worten siehe auch Eigenschaften <code>.style.pageBreakAfter</code> <code>.style.pageBreakBefore</code> <code>.style.wordWrap</code> <code>.style.whiteSpace</code>
<code>.style.wordSpacing</code>	Zusätzlicher Platz zwischen Worten ab IE 6.x
<code>.style.wordWrap</code>	Wortumbruch bei Überschreitung der Objektgrenzen für Block-Elemente wie DIV oder SPAN Elemente müssen kodiert haben <code>.style.height</code> und/oder <code>.style.width</code> <code>.style.position</code> mit "absolute" siehe auch Eigenschaften <code>.style.pageBreakAfter</code> <code>.style.pageBreakBefore</code> <code>.style.wordBreak</code> <code>.style.whiteSpace</code>
<code>.style.writingMode</code>	Flussrichtung bei Objektanzeige wird nicht vererbt ersetzt Eigenschaft <code>.style.layoutFlow</code> , da diese deprecated ist siehe auch Eigenschaft <code>.style.verticalAlign</code>
<code>.style.zIndex</code>	Reihenfolge von überlappten Objekten nur für Elemente mit kodierter Eigenschaft <code>.style.position</code> mit dem Wert "relative" oder "absolute" nicht für Elemente in oder mit Fenster z.B. Control-Objekte mit eigenem Fenster wie Dialogbox im Fenster selektiertes Objekt ab IE 5.5: iframe Objekt unterstützt z-index, da fensterlos siehe auch layer Objekt des Netscape unter 6.x Objekt zoomen mit Layoutkorrektur der Objektumgebung
<code>.style.zoom</code>	

4.3.2.2.4.3.39.19.2. Style-Eigenschaften, die nur per Script ansprechbar sind

Nachfolgende Eigenschaften haben keinen Pendant für das STYLE-Attribut.



.style.cssText	Wert des STYLE-Attributes im Tag des HTML-Elementes
.style.onOffBehavior	deprecated ab IE 5.x
.style.pixelBottom	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und style.posBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.pixelHeight	Objekthöhe identisch mit .style.height und .style.posHeight aber .style.height: String .style.pixelHeight: Integer, nur Pixel .style.posHeight: Float pointing und Integer
.style.pixelLeft	Abstand linker Objektrand zur Umgebung rechte Kante identisch mit Attribut .style.left und .style.posLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer
.style.pixelRight	Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut .style.right und .style.posRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer
.style.pixelTop	Abstand oberer Objektrand zur Umgebung Unterkante identisch mit Attribut .style.top und .style.posTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer
.style.pixelWidth	Objektbreite identisch mit .style.width und .style.posWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer
.style.posBottom	Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und .style.pixelBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.posHeight	Objekthöhe identisch mit .style.height und .style.pixelHeight aber .style.height: String .style.pixelHeight: Integer, nur Pixel .style.posHeight: Float pointing und Integer
.style.position	Art der Objektpositionierung innerhalb Eltern z.B. BODY
.style.posLeft	Abstand linker Objektrand zur Umgebung rechte Kante identisch mit Attribut .style.left und .style.pixelLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer
.style.posRight	Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut .style.right und .style.pixelRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer
.style.posTop	Abstand oberer Objektrand zur Umgebung Unterkante identisch mit Attribut .style.top und .style.pixelTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer
.style.posWidth	Objektbreite identisch mit .style.width und .style.pixelWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer
.style.textDecorationBlink	Wert der Eigenschaft .style.textDecoration auf "blink" prüfen
.style.styleFloat	Text auf blinkend prüfen Textumfluss links, rechts unter IE 5.x: DIV und SPAN müssen .style.width kodiert haben siehe auch Eigenschaften .style.whiteSpace .style.clear .style.float
.style.textDecorationLineThrough	Textdekoration "line-through" (durchstreichen)
.style.textDecorationNone	Textdekoration "none" (deaktivieren aller aktiven Textdekorationen)



.style.textDecorationOverline Textdekoration "overline" (überstreichen)
 .style.textDecorationUnderline Textdekoration "underline" (unterstreichen)
 siehe auch Eigenschaft .style.textUnderlinePosition
 .style.unicodeBidi Bidirektionale Dateneingabe von Unicode in das Element (Zeichenstrom, der vom Elternobjekt in das Element fließt, kann seine eigene Flussrichtung besitzen).

Typische Anwendung ist z.B. der Mix aus Zeichen, die mal von links nach rechts und mal von rechts nach links gerendert werden sollen. Damit ist für den User eine variable Leserichtung möglich, falls der User beide Richtungen benutzen will.

Es ist möglich, den Datenstrom von Zeichen trotz aktiver bidirektionaler Dateneingabe genau nach Vorgabe laut Eigenschaft .style.direction fließen zu lassen.
 Bsp.: Wenn .style.direction auf "ltr" gesetzt ist, so wird diese Richtung konsequent eingehalten, auch wenn der Datenstrom von rechts nach links fließen will.

Die Eigenschaft .style.direction sollte auf "inherit" gesetzt sein, wenn bidirektionaler Datenfluss immer erfolgreich sein soll und die Flussrichtung des Datenstromes unbekannt ist.
 Damit wird die Richtung des Elternobjektes übernommen.

4.3.2.2.4.3.39.19.3. Style-Eigenschaften, die nur per STYLE-Attribut ansprechbar sind

Nachfolgende Eigenschaften haben kein Pendant für die Scriptansteuerung.

float Textumfluss links, rechts
 unter IE 5.x : DIV und SPAN müssen Eigenschaft .style.width besitzen
 siehe auch Eigenschaften .style.whiteSpace
 .style.clear
 .style.styleFloat

4.3.2.2.4.3.39.19.4. Style-Eigenschaften, die nur im HEAD des Dokumentes ansprechbar sind

:active Pseudoklasse des a Objektes für aktiven Link (User hat Link aktiviert)
 ist nicht per Script ansprechbar
 :first Pseudoklasse für @page Regel, siehe auch Objekt page
 :first-letter Pseudoklasse für Style des ersten Buchstaben eines HTML-Elementes (Tags)
 nicht in Script ansprechbar
 nur für Blockelemente wie DIV und SPAN
 Elemente mit Eigenschaft .style.display mit Wert "block"
 ab IE 5.5
 :first-line Pseudoklasse für Style der ersten Zeile eines HTML-Elementes (Tags)
 nicht in Script ansprechbar
 nur für Blockelemente wie DIV und SPAN
 Elemente mit Eigenschaft .style.display mit Wert "block"
 ab IE 5.5
 :hover Pseudoklasse des a Objektes für Link, der nicht aktiviert wurde
 solange Mauszeiger auf dem Link steht, ist die Klasse aktiv
 ist nicht per Script ansprechbar
 :left Pseudoklasse für @page Regel, siehe auch Objekt page
 :link Pseudoklasse des a Objektes für Link, der nicht kürzlich aktiviert wurde.
 ist nicht per Script ansprechbar
 ab IE 4.x
 :right Pseudoklasse für @page Regel, siehe auch Objekt page
 :visited Pseudoklasse des a Objektes für Link, der bereits aktiviert wurde und User ist wieder auf das Dokument zurückgekehrt, das den Link enthält.
 ist nicht per Script ansprechbar

4.3.2.2.4.3.39.19.5. Style-Eigenschaftenübersicht zu Script- und STYLE-Kodierung

Fett markierte Eigenschaften haben kein Pendant für STYLE-Attribut und sind nur per Script ansprechbar.

Fett markierte und unterstrichene Eigenschaften haben kein Pendant für Script und sind nur per STYLE-Attribut ansprechbar.

<u>object.style.eigenschaft</u>	<u>STYLE="..."</u>	<u>Funktion</u>
.media		Media-Typ für Ausgabe eines Objektes
.style.accelerator	ACCELERATOR	ALT+Taste als Kurztastenkombination-Zugriff auf Objekt ein/aus
.style.background	background	Hintergrund eines Objektes bei P, DIV, SPAN: Eigenschaft nur für eingeschlossenen Text
.style.backgroundAttachment	background-attachment	Hintergrundbild nur ab IE 4.x
.style.backgroundColor	background-color	Hintergrundfarbe nur ab IE 4.x Achtung: Für das body Objekt gilt: BGCOLOR Attribut und Eigenschaft .bgColor für die Hintergrundfarbe im BODY sind deprecated und durch <BODY STYLE="background-color:....."> zu ersetzen.
.style.backgroundImage	background-image	Hintergrundbild, das sich über die Hintergrundfarbe legt nur ab IE 4.x



.style.backgroundPosition	background-position	Hintergrund-Position für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundPositionX	background-position-x	Hintergrund-Position X für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundPositionY	background-position-y	Hintergrund-Position Y für Hintergrund z.B. Image (linke obere Ecke)
.style.backgroundRepeat	background-repeat	Kachelung des Hintergrundbildes
.style.behavior	behavior	Ort der Behaviors-Datei *.htc (Verhaltensweise eines Elementes) bzw. Standard-Behavior des IE
.style.border	border	Rahmen alle 4 Seiten: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottom	border-bottom	Rahmen unten: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomColor	border-bottom-color	Rahmen unten: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomStyle	border-bottom-style	Rahmen unten: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderBottomWidth	border-bottom-width	Rahmen unten: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderCollapse	border-collapse	Rahmen alle 4 Seiten: Verschmelzung zum Single Border
.style.borderColor	border-color	Rahmen alle 4 Seiten: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeft	border-left	Rahmen links: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftColor	border-left-color	Rahmen links: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftStyle	border-left-style	Rahmen links: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderLeftWidth	border-left-width	Rahmen links: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRight	border-right	Rahmen rechts: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightColor	border-right-color	Rahmen rechts: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightStyle	border-right-style	Rahmen rechts: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderRightWidth	border-right-width	Rahmen rechts: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderStyle	border-style	Rahmen alle 4 Seiten: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTop	border-top	Rahmen oben: Art und Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopColor	border-top-color	Rahmen oben: Farbe vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopStyle	border-top-style	Rahmen oben: Art vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderTopWidth	border-top-width	Rahmen oben: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.borderWidth	border-width	Rahmen alle 4 Seiten: Dicke vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen
.style.bottom	bottom	Abstand unterer Objektrand zur Umgebung Oberkante Objekt muss position-Eigenschaft kodiert haben identisch mit Attribut .style.pixelBottom und .style.posBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer
.style.clear	clear	Textumfluss links, rechts



		<p>siehe auch Eigenschaften</p> <p>.style.whiteSpace .style.float .style.styleFloat</p>
.style.clip	clip	<p>Position sichtbarer Bereich über dem Objekt</p> <p>Position immer bezüglich Umgebung</p> <p>Objekt muss absolut positioniert sein !</p> <p>ausserhalb des Clipfensters wird alles transparent</p> <p>siehe auch Eigenschaften</p> <p>.style.overflow .style.overflowX .style.overflowY .style.display .style.visibility</p>
.style.color	color	Vordergrundfarbe (Textfarbe)
.style.cssText		Wert des STYLE-Attributes im Tag des HTML-Elementes
.style.cursor	cursor	<p>Mauscursor</p> <p>Arten</p> <p>z.T. erst ab IE 6.x</p>
.style.direction	direction	<p>Horizontale Richtung des Renderns z.B. vom Plain-Textteil eines Elementes, also Leserichtung für den User einstellen (z.B. von rechts nach links)</p> <p>siehe auch Eigenschaft .style.unicodeBidi</p>
.style.display	display	<p>Art der Anzeige (des Renderns) eines Objektes (auch mit Ausrichtung)</p> <p>siehe auch Eigenschaften</p> <p>.style.overflow .style.overflowX .style.overflowY .style.clip .style.visibility</p> <p>z.T. ab IE 6.x</p> <p>Hinweis: Wenn !DOCTYPE am Dokumentanfang kodiert wurde, so darf .style.display nicht auf "list-item" gesetzt sein.</p> <p>Wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde, so kann .style.display auf "list-item" gesetzt sein</p> <p>!DOCTYPE am Dokumenten-Anfang ist erst ab IE 6.x kodierbar !</p>
.style.filter	<p>filter</p> <p><u>float</u></p>	<p>Filter des Internet Explorer implementieren (siehe Objekt filter)</p> <p>Textumfluss links, rechts</p> <p>unter IE 5.x : DIV und SPAN müssen Eigenschaft .style.width besitzen</p> <p>siehe auch Eigenschaften</p> <p>.style.whiteSpace .style.clear .style.styleFloat</p>
.style.font	font	Textfont
.style.fontFamily	font-family	Textfont
.style.fontSize	font-size	Textfont
.style.fontStyle	font-style	Textfont
.style.fontVariant	font-variant	Textfont
.style.fontWeight	font-weight	Textfont
.style.height	height	<p>Höhe Objekt</p> <p>identisch mit .style.pixelHeight und .style.scrollHeight</p> <p>aber .style.height: String</p> <p>.style.pixelHeight: Integer, nur Pixel</p> <p>.style.scrollHeight: Float pointing und Integer</p> <p>Hinweis: Wenn !DOCTYPE am Dokumentanfang kodiert wurde, so kann .style.height auf "auto" gesetzt sein</p> <p>Wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde, so darf .style.height nicht auf "auto" gesetzt sein</p> <p>!DOCTYPE am Dokumenten-Anfang ist erst ab IE 6.x kodierbar !</p>
.style.imeMode	ime-mode	Status des Input Method Editor (IME) für Eingabe von Chinese, Japanese oder Korean Zeichen
.style.layoutFlow	layout-flow	<p>deprecated</p> <p>zu ersetzen durch Eigenschaften .style.writingMode</p> <p>Ost-Asiatische Darstellung (Flussrichtung)</p> <p>siehe auch Eigenschaft .style.verticalAlign für Ausrichtung vertikal für Objekte mit VALIGN-Attribut</p>
.style.layoutGrid	layout-grid	Textzeichen-Layout-Gitter z.B. für asiatische Sprachen
.style.layoutGridChar	layout-grid-char	Textzeichen-Layout-Gitter Zeichengröße



			nur für Blockelemente wie SPAN und DIV etc. verlangt .style.layoutGridMode auf Wert "line" oder "both"
.style.layoutGridLine	layout-grid-line	Textzeichen-Layout-Gitter	Linie
			nur für Blockelemente wie SPAN und DIV etc. verlangt .style.layoutGridMode auf Wert "line" oder "both"
.style.layoutGridMode	layout-grid-mode	Textzeichen-Layout-Gitter	2D
.style.layoutGridType	layout-grid-type	Textzeichen-Layout-Gitter	Typ
			nur für Block-Elemente wie SPAN, DIV
.style.left	left	Abstand linker Objektrand zur Umgebung rechte Kante	
		Objekt muss Eigenschaft .style.position kodiert haben	
		identisch zu .style.pixelLeft	
		aber .style.left:	String
		.style.pixelLeft:	Integer
.style.letterSpacing	letter-spacing	Textzeichen	Abstand
.style.lineBreak	line-break	Zeilenumbruch	
.style.lineHeight	line-height		Abstand zweier Objekte oder zweier Textzeilen ab IE 4.x
.style.listStyle	list-style	Listendarstellung	Aufzählungsliste-Elemente benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStyleImage	list-style-image	Listendarstellung	Marker als Image siehe auch .style.listStyleType benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStylePosition	list-style-position	Listendarstellung	Marker Position benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.listStyleType	list-style-type	Listendarstellung	Markertyp (nicht Image) siehe auch .style.listStyleTypeImage benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten
.style.margin	margin	Aussenrand	Abstand des Aussenrandes links, rechts, oben, unten des Objektes von anderen Objekten im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.marginBottom	margin-bottom	Aussenrand	Abstand unten zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.marginLeft	margin-left	Aussenrand	Abstand links zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.marginRight	margin-right	Aussenrand	Abstand rechts zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.style.marginTop	margin-top	Aussenrand	Abstand oben zum angrenzenden Objekt im Abstandsbereich wird immer transparent angezeigt



		<p>IE 4.x Angaben nicht möglich für TD und TR unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen</p> <p>Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p> <p> Padding: Abstand des Objekthinhaltes zum Aussenrand</p>
.style.minHeight	min-height	<p>minimale Höhe des Objektes</p> <p>ab IE 6.x und nur wenn nicht !DOCTYPE am Dokumentanfang kodiert wurde und dann auch nur für TD, TH und TR innerhalb eines fixierten Tabellenlayouts</p> <p>Tabelle mit fixiertem Layout erzeugen per Eigenschaft</p> <p> .style.tableLayout mit Wert "fixed" wird schneller dargestellt als bei Auto-Layout</p> <p>Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto" also Auto-Layout, also kein fixiertes Tabellenlayout</p>
.style.onOffBehavior		<p>deprecated ab IE 5.x</p> <p> Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound</p>
.style.overflow	overflow	<p>Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster.</p> <p>Anzeige des Objektes mit/ohne Scrollelemente vertikal und horizontal</p> <p>Anzeige des Objektes ein/aus</p> <p>siehe auch Eigenschaften .style.overflowX .style.overflowY .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.overflowX	overflow-x	<p>ab IE 5.x</p> <p>Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster.</p> <p>Anzeige des Objektes mit/ohne Scrollelemente nur horizontal</p> <p>Anzeige des Objektes ein/aus</p> <p>siehe auch Eigenschaften .style.overflow .style.overflowY .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.overflowY	overflow-y	<p>ab IE 5.x</p> <p>Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster, also Überlauf der Anzeige gegenüber Elternfenster.</p> <p>Anzeige des Objektes mit/ohne Scrollelemente nur vertikal</p> <p>Anzeige des Objektes ein/aus</p> <p>siehe auch Eigenschaften .style.overflow .style.overflowX .style.clip .style.display .style.visibility .style.textOverflow</p>
.style.padding	padding	<p>ab IE 5.x</p> <p>Abstand links, rechts, oben, unten zwischen Objekt und Rahmen</p> <p>ab IE 6.x auch für img Objekt</p> <p>unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben</p> <p>Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>
.style.paddingBottom	padding-bottom	<p>Padding: Abstand des Objekthinhaltes zum Aussenrand</p> <p>Abstand unten zwischen Objekt und Rahmen</p> <p>ab IE 6.x auch für img Objekt</p> <p>unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben</p> <p>Margin: Abstand des Aussenrandes eines Objektes zur Umgebung</p>



<code>.style.paddingLeft</code>	<code>padding-left</code>	Padding: Abstand des Objektinhaltes zum Aussenrand Abstand links zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für <code>img</code> Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
<code>.style.paddingRight</code>	<code>padding-right</code>	Padding: Abstand des Objektinhaltes zum Aussenrand Abstand rechts zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für <code>img</code> Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
<code>.style.paddingTop</code>	<code>padding-top</code>	Padding: Abstand des Objektinhaltes zum Aussenrand Abstand oben zwischen Objekt und Margin bzw. Rahmen ab IE 6.x auch für <code>img</code> Objekt unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
<code>.style.pageBreakAfter</code>	<code>page-break-after</code>	Padding: Abstand des Objektinhaltes zum Aussenrand Seitenumbruch nach dem Druck des Objektes beim Druck des Dokumentes nicht möglich für BR und HR-Objekt siehe auch Eigenschaften <code>.style.whiteSpace</code> <code>.style.wordBreak</code> <code>.style.wordWrap</code> <code>.style.pageBreakBefore</code>
<code>.style.pageBreakBefore</code>	<code>page-break-before</code>	Seitenumbruch vor dem Druck des Objektes beim Druck des Dokumentes nicht möglich für BR und HR-Objekt siehe auch Eigenschaften <code>.style.whiteSpace</code> <code>.style.wordBreak</code> <code>.style.wordWrap</code> <code>.style.pageBreakAfter</code>
<code>.style.pixelBottom</code>		Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut <code>.style.bottom</code> und <code>.style.posBottom</code> aber <code>.style.bottom</code> : String <code>.style.pixelBottom</code> : Integer, nur Pixel <code>.style.posBottom</code> : Float pointing und Integer
<code>.style.pixelHeight</code>		Objekthöhe identisch mit <code>.style.height</code> und <code>.style.posHeight</code> aber <code>.style.height</code> : String <code>.style.pixelHeight</code> : Integer, nur Pixel <code>.style.posHeight</code> : Float pointing und Integer
<code>.style.pixelLeft</code>		Abstand linker Objektrand zur Umgebung rechte Kante identisch mit Attribut <code>.style.left</code> und <code>.style.posLeft</code> aber <code>.style.left</code> : String <code>.style.pixelLeft</code> : Integer, nur Pixel <code>.style.posLeft</code> : Float pointing und Integer
<code>.style.pixelRight</code>		Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut <code>.style.right</code> und <code>.style.posRight</code> aber <code>.style.right</code> : String <code>.style.pixelRight</code> : Integer, nur Pixel <code>.style.posRight</code> : Float pointing und Integer
<code>.style.pixelTop</code>		Abstand oberer Objektrand zur Umgebung Unterkante identisch mit Attribut <code>.style.top</code> und <code>.style.posTop</code> aber <code>.style.top</code> : String <code>.style.pixelTop</code> : Integer, nur Pixel <code>.style.posTop</code> : Float pointing und Integer
<code>.style.pixelWidth</code>		Objektbreite identisch mit <code>.style.width</code> und <code>.style.posWidth</code> aber <code>.style.width</code> : String <code>.style.pixelWidth</code> : Integer, nur Pixel <code>.style.posWidth</code> : Float pointing und Integer
<code>.style.posBottom</code>		Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut <code>.style.bottom</code> und <code>.style.pixelBottom</code> aber <code>.style.bottom</code> : String <code>.style.pixelBottom</code> : Integer, nur Pixel <code>.style.posBottom</code> : Float pointing und Integer
<code>.style.posHeight</code>		Objekthöhe identisch mit <code>.style.height</code> und <code>.style.pixelHeight</code> aber <code>.style.height</code> : String <code>.style.pixelHeight</code> : Integer, nur Pixel



.style.position		Art der Objektpositionierung innerhalb Eltern z.B. BODY
.style.posLeft		Abstand linker Objektrand zur Umgebung rechte Kante identisch mit Attribut .style.left und .style.pixelLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer
.style.posRight		Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut .style.rigth und .style.pixelRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer
.style.posTop		Abstand oberer Objektrand zur Umgebung Unterkante identisch mit Attribut .style.top und .style.pixelTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer
.style.posWidth		Objektbreite identisch mit .style.width und .style.pixelWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer
.style.rubyAlign	ruby-align	Ausrichtung des ruby Objektes hier nicht weiter erklärt
.style.rubyOverhang	ruby-overhang	Überhang des ruby Objektes hier nicht weiter erklärt
.style.rubyPosition	ruby-position	Position des ruby Objektes hier nicht weiter erklärt
.style.right	right	Abstand rechter Objektrand zur Umgebung linke Kante benötigt kodiertes .style.position Attribut identisch mit Attribut .style.posRigth und .style.pixelRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer
.style.scrollbar3dLightColor	scrollbar-3dlight-color	Farbe der Kante von Scrollbalken-Box (in der der Scrollbalken wandert) und Scroll-Pfeil-Box auf den Enden des Scrollbalken
.style.scrollbarArrowColor	scrollbar-arrow-color	Farbe Pfeil auf den Enden des Scrollbalken
.style.scrollbarBaseColor	scrollbar-base-color	Farbe aller Scrollbalken-Elemente
.style.scrollbarDarkShadowColor	scrollbar-darkshadow-color	Farbe der Rinne des Scrollbalkens
.style.scrollbarFaceColor	scrollbar-face-color	Farbe Scrollbild und Scrollpfeile des Scrollbalkens
.style.scrollbarHighlightColor	scrollbar-highlight-color	Farbe der oberen und linken Kante der Scrollbox und der Scrollpfeile des Scrollbalkens
.style.scrollbarShadowColor	scrollbar-shadow-color	Farbe der unteren und rechten Kante der Scrollbox und der Pfeile des Scrollbalkens
.style.scrollbarTrackColor	scrollbar-track-color	Farbe des Trackelementes von Scrollbalken (Track = ziehen)
.style.styleFloat		Textumfluss links, rechts unter IE 5.x: DIV und SPAN müssen .style.width kodiert haben siehe auch Eigenschaften .style.whiteSpace .style.clear .style.float
.style.tableLayout	table-layout	Tabellen-Layout (auto oder fixed) siehe auch Eigenschaft .style.minHeight Tabelle mit fixiertem Layout erzeugen per Eigenschaft .style.tableLayout mit Wert "fixed" wird schneller dargestellt als bei Auto-Layout Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto" also Auto-Layout, also kein fixiertes Tabellenlayout
.style.textAlign	text-align	Textausrichtung nur für Block-Elemente wie SPAN oder DIV siehe auch .style.textAlignLast und .style.textJustify
.style.textAlignLast	text-align-last	Textausrichtung der letzten Zeile nur für Block-Elemente wie SPAN oder DIV siehe auch .style.textAlign und .style.textJustify
.style.textAutospace	text-autospace	Zusatz-Textabstand bei asiatischen Zeichen
.style.textDecoration	text-decoration	dekoratives Layout eines Textes für nichtleere Textobjekte (unterstreichen, überstreichen, durchstreichen) z.B. ist ein leeres bei Block-Element: Layout wird an Kinder vererbt siehe auch Eigenschaft .style.textDecorationBlink
.style.textDecorationBlink		Wert der Eigenschaft .style.textDecoration auf "blink" prüfen Text auf blinkend prüfen
.style.textDecorationLineThrough		Textdekoration "line-through" (durchstreichen)



.style.textDecorationNone		Textdekoration "none" (deaktivieren aller aktiven Textdekorationen)
.style.textDecorationOverline		Textdekoration "overline" (überstreichen)
.style.textDecorationUnderline		Textdekoration "underline" (unterstreichen) siehe auch Eigenschaft .style.textUnderlinePosition
.style.textIndent	text-indent	Position-Ident in der der ERSTEN Zeile eines Textes Ident nicht möglich in der Mitte eines umgebrochenen Objekts (Umbrechung z.B. per BR)
.style.textJustify	text-justify	Textausrichtung Blocksatz nur für Blockelemente wie SPAN oder DIV Attribut .style.textAlign muss Wert "justify" haben
.textKashidaSpace	text-kashida-space	Typographischer Effekt "Kashida" für Arabisch siehe auch Eigenschaft .style.textJustify nur für Blockelemente wie DIV und SPAN
.style.textOverflow	text-overflow	Rahmen um Text ab IE 6.x Eigenschaft ermöglicht einen Rahmen um einen Text, wobei der Text im Rahmen überlaufen kann Überlauf und Rahmen können folgende Formen haben: Abschneiden an Rahmengrenze Abschneiden an Rahmengrenze mit automatischem Anfügen von "..." als Andeutung einer Fortsetzung kein Abschneiden, also Rahmen in voller Textbreite Achtung: Damit der Text nicht umgebrochen wird im Rahmen, muss die Eigenschaft .style.whiteSpace auf "nowrap" gesetzt sein bzw. der Text in das NOBR-Tag eingeschlossen sein. Desweiteren sollte die Eigenschaft .style.overflow auf "hidden" gesetzt sein, damit keine Scrollbalken erscheinen (im Gegensatz zu "auto" oder "scroll")
.style.textTransform	text-transform	Textkonvertierung nach Gross oder Klein oder erstes Zeichen im Wort stets groß siehe auch Objekt String
.style.textUnderlinePosition	text-underline-position	Position der Textdekoration "underline" (unterstreichen) siehe auch Eigenschaft .style.textDecorationUnderline
.style.top	top	Abstand oberer Objektrand zur Umgebung Unterkante benötigt kodierte Eigenschaft .style.position identisch mit Attribut .style.posTop und style.pixelTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer
.style.unicodeBidi	unicode-bidi	Bidirektionale Dateneingabe von Unicode in das Element (Zeichenstrom, der vom Elternobjekt in das Element fließt, kann seine eigene Flussrichtung besitzen). Typische Anwendung ist z.B. der Mix aus Zeichen, die mal von links nach rechts und mal von rechts nach links gerendert werden sollen. Damit ist für den User eine variable Leserichtung möglich, falls der User beide Richtungen benutzen will. Es ist möglich, den Datenstrom von Zeichen trotz aktiver bidirektionaler Dateneingabe genau nach Vorgabe laut Eigenschaft .style.direction fließen zu lassen. Bsp.: Wenn .style.direction auf "ltr" gesetzt ist, so wird diese Richtung konsequent eingehalten, auch wenn der Datenstrom von rechts nach links fließen will. Die Eigenschaft .style.direction sollte auf "inherit" gesetzt sein, wenn bidirektionaler Datenfluss immer erfolgreich sein soll und die Flussrichtung des Datenstromes unbekannt ist. Damit wird die Richtung des Elternobjektes übernommen.
.style.verticalAlign	vertical-align	Ausrichtung vertikal für Objekte mit VALIGN-Attribut und bei Textobjekten ohne VALIGN-Attribut
.style.visibility	visibility	Objekt-Sichtbarkeit ohne Reservierung des Platzs im Layout der Umgebung ab IE 5.x Sichtbarkeit eines Kindobjektes, auch wenn Eltern unsichtbar sind siehe auch Eigenschaften .style.overflow .style.overflowX .style.overflowY .style.clip .style.display



		Hinweis: display:none Objekt behält den Platz im Layout der Umgebung, obwohl Objekt unsichtbar ist
.style.whiteSpace	white-space	automatischer Zeilenumbruch bei Block-Elementen wie DIV und SPAN Umbrechen erfolgt an den Stellen wo Blanks, Tabs stehen, aber nicht an der Stelle eines geschützten Blanks () siehe auch Eigenschaften .style.pageBreakAfter .style.pageBreakBefore .style.wordBreak .style.wordWrap
.style.width	width	Objektbreite identisch mit Attribut .style.posWidth und .style.pixelWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer
.style.wordBreak	word-break	Zeilenumbruch in Worten siehe auch Eigenschaften .style.pageBreakAfter .style.pageBreakBefore .style.wordWrap .style.whiteSpace
.style.wordSpacing	word-spacing	Zusätzlicher Platz zwischen Worten ab IE 6.x
.style.wordWrap	word-wrap	Wortumbruch bei Überschreitung der Objektgrenzen für Block-Elemente wie DIV oder SPAN Elemente müssen kodiert haben .style.height und/oder .style.width .style.position mit "absolute" siehe auch Eigenschaften .style.pageBreakAfter .style.pageBreakBefore .style.wordBreak .style.whiteSpace
.style.writingMode	writing-mode	Flussrichtung bei Objektanzeige wird nicht vererbt ersetzt Eigenschaft .style.layoutFlow, da diese deprecated ist siehe auch Eigenschaft .style.verticalAlign
.style.zIndex	z-index	Reihenfolge von überlappten Objekten nur für Elemente mit kodierter Eigenschaft .style.position mit dem Wert "relative" oder "absolute" nicht für Elemente in oder mit Fenster z.B. Control-Objekte mit eigenem Fenster wie Dialogbox im Fenster selektiertes Objekt ab IE 5.5: iframe Objekt unterstützt z-index, da fensterlos siehe auch layer Objekt des Netscape unter 6.x
.style.zoom	zoom	Objekt zoomen mit Layoutkorrektur der Objektumgebung
.type		Sprache des Cascading Style Sheets (CSS) des style Objektes

4.3.2.2.4.3.39.20. Methoden

4.3.2.2.4.3.39.20.1. Methoden des DOM

Das style Objekt besitzt - bis auf 1 Ausnahme- keine eigenen Methoden. Die Objekte selbst müssen Methoden besitzen, um Style manipulieren zu können. Nachfolgende Methoden sind nicht immer im Objekt implementiert (siehe Objektbeschreibung).

.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.createAttribute()	ein Attribut im Dokument erzeugen und Referenz auf das erzeugte Attribut liefern Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! DOM nicht geändert
.createElement()	HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern Achtung: Erzeugtes Objekt muss in DOM noch per Methode .insertBefore() bzw. .appendChild() eingereiht werden. Hinweis: Attribute mit der Methode .createAttribute() erzeugen
.createStyleSheet()	DOM wird geändert Style-Sheet-Objekt im Dokument erzeugen und Referenz liefern DOM wird geändert Beispiel: document.createStyleSheet('styles.css');
.expression()	Wert einer Style-Eigenschaft per STYLE-Attribut-Wert in HTML als Ausdruck definieren für spätere Berechnung per Methode .getExpression() Ausdruck nur als Script kodierbar DOM wird geändert siehe auch Methode .setExpression()



.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getElementById()	Referenz auf das im Dokument ZUERST gefundene Objekt laut ID (analog zum ID-Attribut) liefern Achtung: Objekte, die kein ID besitzen, werden nicht erfasst ! Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode getElementsByName() Für Verwaltung per Tag-Name siehe Methode getElementsByTagName() wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenziert Eine Referenzierung einer Collection der Objekte mit gemeinsamen ID ist leider nicht möglich. Deswegen der strenge Hinweis: In der Regel werden ID vom Programmierer objektweise getrennt vergeben, es sei denn, man will bewusst eine Gruppe von Objekten (z.B. RadioBox) verwaltbar machen und kennt diese Objekte. Die maschinelle Analyse eines fremden Dokumentes mit der Methode getElementById() ist nicht möglich.
.getElementsByName()	DOM nicht geändert Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME (analog zum Attribut NAME) liefern Achtung: Objekte, die kein NAME besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per Tag-Name siehe Methode getElementsByTagName()
.getElementsByTagName()	DOM nicht geändert Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.insertAdjacentHTML()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden
.insertAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.pasteHTML()	DOM wird geändert Textbereich-Inhalt durch Text ersetzen oder leeren Textbereich füllen Text kann auch HTML enthalten Tabelle nur mit kompletten HTML-Code einfügbar, also z.B. keine einzelne Zelle Achtung: Wenn HTML-Code im Text enthalten ist, muss das Elternobjekt auch diesen ansich unterstützen Bsp.: textArea erlaubt kein HTML-Code HTML-Code wird geparkt per textrange Objekt nur unter Windows 32-Bit
.recalc()	dynamischen Eigenschaften des Dokumentes neu berechnen Hinweis: andere Objekte, die Eigenschaften des Dokumentes nutzen, werden auch neu berechnet, wenn Eigenschaften nicht in einem Berechnungsausdruck vorliegen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der



Form objekt.style.eigenschaft. dient.
 Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
 DOM wird nicht geändert
 .setAttribute() Wert von vorhandenem Attribut setzen
 wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
 DOM wird nur bei Erzeugung geändert
 .setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert

4.3.2.2.4.3.39.20.2. Konvertierungsmethode

Style besitzt für die style-gerechte Umwandlung eines Url-Wertes die Methode url(), welche nicht direkt als Methode des Style-Objektes implementiert ist und damit ohne Punktreferenz kodiert wird (vermutlich Parser-interne Funktion).

url()

Beispiele für Bild:

```
<STYLE>
    .style1 {background:beige url(sphere.jpg) no-repeat top center}
    .style2 {background:ivory url(sphere.jpeg) no-repeat bottom right}
</STYLE>

<SPAN onclick="this.style.background='beige url(sphere.jpeg) no-repeat top center'"></SPAN>

<STYLE>
    .setUrl { background-image: url(sphere.jpg) }
    .loseUrl { background-image: url(none) }
</STYLE>
<SPAN STYLE="font-size:14"
    onmouseover="this.className='setUrl'"
    onmouseout="this.className='loseUrl'"
>
</SPAN>
```

Beispiele für Behavior:

```
url(#objID)          mit objID als ID des OBJECT-Tags
url(#default#behaviorName) eines Standard-Behaviors des IE

STYLE="behavior:url(a1.htc) url(a2.htc)"

<STYLE>
    .Klasse { behavior:url(#myObject) }
</STYLE>

<STYLE>
    DIV { behavior:url(fly.htc) url (zoom.htc) url (fade.htc)}
</STYLE>
```

Beispiel für Cursorformen aus **Datei** und **nicht** für im Browser implementierte Standard-Cursorformen:
 url('mycursor.cur')

Hinweise: alert (zeiger_auf_objekt.style.cursor); liefert beim IE den String 'url(cursor_form)'
 mit cursor_form für den aktuellen **und** gesetzten Cursor.

style.cursor ist standardgemäß mit einer Leerkette belegt, solange **kein** Cursor gesetzt wird
 kann nicht mit url(cursor_form) belegt werden, wenn es sich um eine **standardgemäß im Browser implementierte** Cursorform handelt wie z.B. 'hand' oder 'normal'.
 Grund: Diese Cursorformen sind keine Dateien, obwohl alert den String 'url(cursor_form)' anzeigt.

4.3.2.2.4.3.39.21. Objekte (Auswahl) und ihr Style-Eigenschaften

Nachfolgende Beschreibungen nennen nur die per Script ansprechbaren Bezeichner der Style-Eigenschaften, es sei denn, die Eigenschaft ist **nur** per HTML und STYLE-Attribut ansprechbar.

Für in HTML per STYLE-Attribut kodierbare Bezeichner der Style-Eigenschaften: siehe Beschreibung der jeweiligen Eigenschaft.

Fettmarkierte Eigenschaften sind nur per STYLE-Attribut kodierbar.

a Objekt

:active
 :hover

:link
 :visited

accelerator
 background



backgroundAttachment
background-color
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color

cursor
direction
display
filter
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight

pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
text-transform
textUnderlinePosition
top
unicodeBidi
visibility
width
wordSpacing
wordWrap
writingMode
zIndex
zoom

applet Objekt

accelerator
backgroundPositionX
backgroundPositionY
behavior
bottom
clear
clip
color
cursor
display

FLOAT

fontSize
height
layoutGrid
layoutGridMode
left
overflow
overflowX
overflowY

padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right

scrollbar3dLightColor
scrollbarArrowColor
scrollbarBaseColor
scrollbarDarkShadowColor
scrollbarFaceColor
scrollbarHighlightColor
scrollbarShadowColor
scrollbarTrackColor
styleFloat
textAutospace
textUnderlinePosition
top
visibility
width
wordWrap
zIndex
zoom

area Objekt

behavior

bg sound Objekt

behavior

textAutospace

textUnderlinePosition

body Objekt

:first-letter
:first-line
accelerator
background
backgroundAttachment
background-color
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border

borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth

borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
color
cursor
direction
display
filter
font
fontFamily



fontSize	paddingRight	scrollbarTrackColor
fontStyle	paddingTop	textAlign
fontVariant	pageBreakAfter	textAlignLast
fontWeight	pageBreakBefore	textAutospace
layoutGrid	pixelBottom	textDecoration
layoutGridChar	pixelHeight	textDecorationBlink
layoutGridLine	pixelLeft	textDecorationLineThrough
layoutGridMode	pixelRight	textDecorationNone
layoutGridType	pixelTop	textDecorationOverline
letterSpacing	pixelWidth	textDecorationUnderline
lineBreak	posBottom	textIndent
lineHeight	posHeight	textJustify
margin	posLeft	textKashidaSpace
marginBottom	posRight	textOverflow
marginLeft	posTop	textTransform
marginRight	posWidth	textUnderlinePosition
marginTop	scrollbar3dLightColor	unicodeBidi
overflow	scrollbarArrowColor	visibility
overflowX	scrollbarBaseColor	whiteSpace
overflowY	scrollbarDarkShadowColor	wordBreak
padding	scrollbarFaceColor	wordSpacing
paddingBottom	scrollbarHighlightColor	wordWrap
paddingLeft	scrollbarShadowColor	zoom

button Objekt

background	color	pixelRight
backgroundAttachment	direction	pixelTop
backgroundColor	display	pixelWidth
backgroundImage	filter	posBottom
backgroundPosition	float	posHeight
backgroundPositionX	font	position
backgroundPositionY	fontFamily	posLeft
backgroundRepeat	fontSize	posRight
behavior	fontStyle	posTop
border	fontVariant	posWidth
borderBottom	fontWeight	right
borderBottomColor	height	styleFloat
borderBottomStyle	layoutFlow	textAutospace
borderBottomWidth	layoutGrid	textDecoration
borderColor	layoutGridMode	textDecorationBlink
borderLeft	left	textDecorationLineThrough
borderLeftColor	letterSpacing	textDecorationNone
borderLeftStyle	lineHeight	textDecorationOverline
borderLeftWidth	margin	textDecorationUnderline
borderRight	marginBottom	textIndent
borderRightColor	marginLeft	textOverflow
borderRightStyle	marginRight	textTransform
borderRightWidth	marginTop	textUnderlinePosition
borderStyle	padding	top
borderTop	paddingBottom	unicodeBidi
borderTopColor	paddingLeft	visibility
borderTopStyle	paddingRight	width
borderTopWidth	paddingTop	wordSpacing
borderWidth	pageBreakAfter	wordWrap
bottom	pixelBottom	writingMode
clear	pixelHeight	zIndex
clip	pixelLeft	zoom

comment Objekt

backgroundPositionX	pixelRight	posRight
backgroundPositionY	pixelTop	posTop
behavior	pixelWidth	posWidth
pixelBottom	posBottom	textAutospace
pixelHeight	posHeight	textUnderlinePosition
pixelLeft	posLeft	

currentStyle Objekt

alle Style-Eigenschaften des style Objektes (auch die Pseudoklassen), die aber **nur** lesbar sind
 Natürlich sind in `zeiger_auf_objekt.currentStyle.style_eigenschaft` nur die wirklich im Objekt implementierten Style-Eigenschaften referenziert.

custom Objekt

accelerator	backgroundAttachment	backgroundImage
background	backgroundColor	backgroundPosition



backgroundPositionX	fontSize	posTop
backgroundPositionY	fontStyle	posWidth
backgroundRepeat	fontVariant	scrollbar3dLightColor
behavior	fontWeight	scrollbarArrowColor
border	height	scrollbarBaseColor
borderBottom	layoutFlow	scrollbarDarkShadowColor
borderBottomColor	layoutGrid	scrollbarFaceColor
borderBottomStyle	layoutGridMode	scrollbarHighlightColor
borderBottomWidth	left	scrollbarShadowColor
borderColor	letterSpacing	scrollbarTrackColor
borderLeft	lineHeight	styleFloat
borderLeftColor	margin	textAlignLast
borderLeftStyle	marginBottom	textAutospace
borderLeftWidth	marginLeft	textDecoration
borderRight	marginRight	textDecorationBlink
borderRightColor	marginTop	textDecorationLineThrough
borderRightStyle	overflow	textDecorationNone
borderRightWidth	overflowX	textDecorationOverline
borderStyle	overflowY	textDecorationUnderline
borderTop	padding	textKashidaSpace
borderTopColor	paddingBottom	textOverflow
borderTopStyle	paddingLeft	textTransform
borderTopWidth	paddingRight	textUnderlinePosition
borderWidth	paddingTop	top
bottom	pixelBottom	unicodeBidi
clear	pixelHeight	verticalAlign
clip	pixelLeft	visibility
color	pixelRight	whiteSpace
cursor	pixelTop	width
direction	pixelWidth	wordSpacing
display	posBottom	wordWrap
filter	posHeight	writingMode
float	position	zIndex
font	posLeft	zoom
fontFamily	posRight	

div Objekt

:first-letter	direction	pixelHeight
:first-line	display	pixelLeft
accelerator	filter	pixelRight
background	float	pixelTop
backgroundAttachment	font	pixelWidth
backgroundColor	fontFamily	posBottom
backgroundImage	fontSize	posHeight
backgroundPosition	fontStyle	position
backgroundPositionX	fontVariant	posLeft
backgroundPositionY	fontWeight	posRight
backgroundRepeat	height	posTop
behavior	layoutFlow	posWidth
border	layoutGrid	right
borderBottom	layoutGridChar	scrollbar3dLightColor
borderBottomColor	layoutGridLine	scrollbarArrowColor
borderBottomStyle	layoutGridMode	scrollbarBaseColor
borderBottomWidth	layoutGridType	scrollbarDarkShadowColor
borderColor	left	scrollbarFaceColor
borderLeft	letterSpacing	scrollbarHighlightColor
borderLeftColor	lineBreak	scrollbarShadowColor
borderLeftStyle	lineHeight	scrollbarTrackColor
borderLeftWidth	margin	styleFloat
borderRight	marginBottom	textAlign
borderRightColor	marginLeft	textAlignLast
borderRightStyle	marginRight	textAutospace
borderRightWidth	marginTop	textDecoration
borderStyle	overflow	textDecorationBlink
borderTop	overflowX	textDecorationLineThrough
borderTopColor	overflowY	textDecorationNone
borderTopStyle	padding	textDecorationOverline
borderTopWidth	paddingBottom	textDecorationUnderline
borderWidth	paddingLeft	textIndent
bottom	paddingRight	textJustify
clear	paddingTop	textKashidaSpace
clip	pageBreakAfter	textOverflow
color	pageBreakBefore	textTransform
cursor	pixelBottom	textUnderlinePosition



top
 unicodeBidi
 visibility
 whiteSpace

width
 wordBreak
 wordSpacing
 wordWrap

writingMode
 zIndex
 zoom

document Objekt

keine

embed Objekt

accelerator
 backgroundPositionX
 backgroundPositionY
 behavior
 border
 borderBottom
 borderBottomColor
 borderBottomStyle
 borderBottomWidth
 borderColor
 borderLeft
 borderLeftColor
 borderLeftStyle
 borderLeftWidth
 borderRight
 borderRightColor
 borderRightStyle
 borderRightWidth
 borderStyle
 borderTop
 borderTopColor
 borderTopStyle
 borderTopWidth
 borderWidth
 clear
 clip

cursor
 direction
 display
float
 height
 layoutGrid
 layoutGridMode
 left
 margin
 marginBottom
 marginLeft
 marginRight
 marginTop
 overflow
 overflowX
 overflowY
 padding
 paddingBottom
 paddingLeft
 paddingRight
 paddingTop
 pixelBottom
 pixelHeight
 pixelLeft
 pixelRight
 pixelTop

pixelWidth
 posBottom
 posHeight
 position
 posLeft
 posRight
 posTop
 posWidth
 scrollbar3dLightColor
 scrollbarArrowColor
 scrollbarBaseColor
 scrollbarDarkShadowColor
 scrollbarFaceColor
 scrollbarHighlightColor
 scrollbarShadowColor
 scrollbarTrackColor
 styleFloat
 textAutospace
 textUnderlinePosition
 top
 unicodeBidi
 visibility
 wordWrap
 zoom

event Objekt

kein

fieldset Objekt

:first-letter
 :first-line
 accelerator
 background
 backgroundAttachment
 backgroundColor
 backgroundImage
 backgroundPosition
 backgroundPositionX
 backgroundPositionY
 backgroundRepeat
 behavior
 border
 borderBottom
 borderBottomColor
 borderBottomStyle
 borderBottomWidth
 borderColor
 borderLeft
 borderLeftColor
 borderLeftStyle
 borderLeftWidth
 borderRight
 borderRightColor
 borderRightStyle
 borderRightWidth
 borderStyle
 borderTop
 borderTopColor
 borderTopStyle
 borderTopWidth
 borderWidth
 bottom
 clear

clip
 color
 cursor
 direction
 display
 filter
float
 font
 fontFamily
 fontSize
 fontStyle
 fontVariant
 fontWeight
 height
 layoutFlow
 layoutGrid
 layoutGridChar
 layoutGridLine
 layoutGridMode
 layoutGridType
 left
 letterSpacing
 lineBreak
 lineHeight
 margin
 marginBottom
 marginLeft
 marginRight
 marginTop
 overflow
 overflowX
 overflowY
 padding
 paddingBottom

paddingLeft
 paddingRight
 paddingTop
 pageBreakAfter
 pageBreakBefore
 pixelBottom
 pixelHeight
 pixelLeft
 pixelRight
 pixelTop
 pixelWidth
 posBottom
 posHeight
 position
 posLeft
 posRight
 posTop
 posWidth
 right
 styleFloat
 textAlign
 textAlignLast
 textAutospace
 textDecoration
 textDecorationBlink
 textDecorationLineThrough
 textDecorationNone
 textDecorationOverline
 textDecorationUnderline
 textIndent
 textJustify
 textKashidaSpace
 textOverflow
 textTransform



textUnderlinePosition
top
unicodeBidi
visibility

whiteSpace Sets
width
wordBreak
wordSpacing

wordWrap
writingMode
zIndex
zoom

font Objekt

accelerator
backgroundPositionX
backgroundPositionY
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth

bottom
direction
display
filter
height
layoutFlow
layoutGrid
layoutGridMode
left
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight

pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
textAutospace
textOverflow
textUnderlinePosition
top
unicodeBidi
whiteSpace
width
wordWrap
writingMode
zoom

form Objekt

:first-letter
:first-line
accelerator
background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color
cursor
direction
display

filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridChar
layoutGridLine
layoutGridMode
layoutGridType
left
letterSpacing
lineBreak
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop

pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
styleFloat
textAlign
textAlignLast
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textIndent
textJustify
textKashidaSpace
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
whiteSpace
width
width
wordBreak
wordSpacing
wordWrap
writingMode
zIndex
zoom

frame Objekt

backgroundPositionX
backgroundPositionY
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle

borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
display
filter
height
layoutGrid
layoutGridMode
padding
paddingBottom
paddingLeft
paddingRight

paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight
posTop
posWidth
textAutospace
zoom

frameset Objekt

backgroundPositionX
backgroundPositionY
behavior
borderBottom
borderLeft
borderRight
borderTop
filter
height

layoutGrid
layoutGridMode
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom

posHeight
posLeft
posRight
posTop
posWidth
textUnderlinePosition
zoom

head Objekt

backgroundPositionX
backgroundPositionY
behavior
layoutGrid
layoutGridMode
pixelBottom
pixelHeight

pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft

posRight
posTop
posWidth
textAutospace
textUnderlinePosition
zoom



html Objekt

background	fontFamily	textDecoration
backgroundAttachment	fontSize	textDecorationBlink
backgroundColor	fontStyle	textDecorationLineThrough
backgroundImage	fontVariant	textDecorationNone
backgroundPosition	fontWeight	textDecorationOverline
backgroundRepeat	letterSpacing	textDecorationUnderline
behavior	lineHeight	textTransform
color	overflow	textUnderlinePosition
cursor	overflowX	visibility
display	overflowY	wordSpacing
font	textAutospace	wordWrap

html comment Objekt

keine

iframe Objekt

accelerator	borderTopColor	overflowY
backgroundAttachment	borderTopStyle	pixelBottom
backgroundColor	borderTopWidth	pixelHeight
backgroundPositionX	borderWidth	pixelLeft
backgroundPositionY	bottom	pixelRight
behavior	clear	pixelTop
borderBottom	clip	pixelWidth
borderBottomColor	cursor	posBottom
borderBottomStyle	display	posHeight
borderBottomWidth	filter	position
borderColor	float	posLeft
borderLeft	height	posRight
borderLeftColor	layoutGrid	posTop
borderLeftStyle	layoutGridMode	posWidth
borderLeftWidth	left	right
borderRight	margin	styleFloat
borderRightColor	marginBottom	textAutospace
borderRightStyle	marginLeft	top
borderRightWidth	marginRight	visibility
borderStyle	marginTop	zIndex
borderTop	overflowX	zoom

img Objekt

background	borderWidth	paddingRight
backgroundAttachment	bottom	paddingTop
backgroundColor	clear	pixelBottom
backgroundImage	clip	pixelHeight
backgroundPosition	cursor	pixelLeft
backgroundPositionX	direction	pixelRight
backgroundPositionY	display	pixelTop
backgroundRepeat	filter	pixelWidth
behavior	float	posBottom
border	font	posHeight
borderBottom	fontFamily	position
borderBottomColor	fontStyle	posLeft
borderBottomStyle	fontVariant	posRight
borderBottomWidth	fontWeight	posTop
borderColor	height	posWidth
borderLeft	layoutGrid	right
borderLeftColor	layoutGridMode	styleFloat
borderLeftStyle	left	textAutospace
borderLeftWidth	letterSpacing	textUnderlinePosition
borderRight	lineHeight	top
borderRightColor	margin	unicodeBidi
borderRightStyle	marginBottom	verticalAlign
borderRightWidth	marginLeft	visibility
borderStyle	marginRight	wordSpacing
borderTop	marginTop	wordWrap
borderTopColor	padding	zoom
borderTopStyle	paddingBottom	
borderTopWidth	paddingLeft	

input Objekt

layoutFlow	wordWrap	zoom
textUnderlinePosition	writingMode	

input button Objekt

background	backgroundAttachment	backgroundColor
------------	----------------------	-----------------



backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color
cursor

direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom

pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
wordSpacing
wordWrap
writingMode
zIndex
zoom

input checkbox Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear

clip
color
cursor
direction
display
filter
font
float
fontFamily
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight

paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
styleFloat
textAutospace
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
unicodeBidi
visibility
wordSpacing
wordWrap
zIndex
zoom

input file Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior

border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle

borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle



borderTopWidth
borderWidth
bottom
clear
clip
color
cursor
direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing

input hidden Objekt

behavior
layoutFlow

input image Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip

input password Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor

lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position

textAutospace
textUnderlinePosition

color
cursor
direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop

borderBottomStyle
borderBottomWidth
borderColor
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle

posLeft
posRight
posTop
posWidth
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
wordSpacing
wordWrap
writingMode
zIndex
zoom

wordWrap
zoom

pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
styleFloat
textAutospace
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
wordSpacing
wordWrap
zIndex
zoom

borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color
cursor
direction
display



filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow

input radio Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip

input reset Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle

overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right

color
cursor
direction
display
filter
float
font
fontFamily
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom

borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color
cursor
direction
display

styleFloat
textAlign
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
wordSpacing
wordWrap
writingMode
zIndex
zoom

pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
wordSpacing
wordWrap
zIndex
zoom

filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft



marginRight	pixelWidth	textDecorationOverline
marginTop	posBottom	textDecorationUnderline
overflow	posHeight	textOverflow
overflowX	position	textTransform
overflowY	posLeft	textUnderlinePosition
padding	posRight	top
paddingBottom	posTop	unicodeBidi
paddingLeft	posWidth	visibility
paddingRight	right	wordSpacing
paddingTop	styleFloat	wordWrap
pixelBottom	textAutospace	writingMode
pixelHeight	textDecoration	zIndex
pixelLeft	textDecorationBlink	zoom
pixelRight	textDecorationLineThrough	
pixelTop	textDecorationNone	

input submit Objekt

background	cursor	pixelLeft
backgroundAttachment	direction	pixelRight
backgroundColor	display	pixelTop
backgroundImage	filter	pixelWidth
backgroundPosition	float	posBottom
backgroundPositionX	font	posHeight
backgroundPositionY	fontFamily	position
backgroundRepeat	fontSize	posLeft
behavior	fontStyle	posRight
border	fontVariant	posTop
borderBottom	fontWeight	posWidth
borderBottomColor	height	right
borderBottomStyle	layoutFlow	styleFloat
borderBottomWidth	layoutGrid	textAutospace
borderColor	layoutGridMode	textDecoration
borderLeft	left	textDecorationBlink
borderLeftColor	letterSpacing	textDecorationLineThrough
borderLeftStyle	lineHeight	textDecorationNone
borderLeftWidth	margin	textDecorationOverline
borderRight	marginBottom	textDecorationUnderline
borderRightColor	marginLeft	textOverflow
borderRightStyle	marginRight	textTransform
borderRightWidth	marginTop	textUnderlinePosition
borderStyle	overflow	top
borderTop	overflowX	unicodeBidi
borderTopColor	overflowY	visibility
borderTopStyle	padding	wordSpacing
borderTopWidth	paddingBottom	wordWrap
borderWidth	paddingLeft	writingMode
bottom	paddingRight	zIndex
clear	paddingTop	zoom
clip	pixelBottom	
color	pixelHeight	

input text Objekt

background	borderLeftColor	color
backgroundAttachment	borderLeftStyle	cursor
backgroundColor	borderLeftWidth	direction
backgroundImage	borderRight	display
backgroundPosition	borderRightColor	filter
backgroundPositionX	borderRightStyle	font
backgroundPositionY	borderRightWidth	fontFamily
backgroundRepeat	borderStyle	fontSize
behavior	borderTop	fontStyle
border	borderTopColor	fontVariant
borderBottom	borderTopStyle	fontWeight
borderBottomColor	borderTopWidth	height
borderBottomStyle	borderWidth	imeMode
borderBottomWidth	bottom	layoutFlow
borderColor	clear	
borderLeft	clip	

label Objekt

accelerator	backgroundPosition	border
background	backgroundPositionX	borderBottom
backgroundAttachment	backgroundPositionY	borderBottomColor
backgroundColor	backgroundRepeat	borderBottomStyle
backgroundImage	behavior	borderBottomWidth



borderColor	fontWeight	position
borderLeft	height	posLeft
borderLeftColor	layoutFlow	posRight
borderLeftStyle	layoutGrid	posTop
borderLeftWidth	layoutGridMode	posWidth
borderRight	left	right
borderRightColor	letterSpacing	styleFloat
borderRightStyle	lineHeight	textAutospace
borderRightWidth	margin	textDecoration
borderStyle	marginBottom	textDecorationBlink
borderTop	marginLeft	textDecorationLineThrough
borderTopColor	marginRight	textDecorationNone
borderTopStyle	marginTop	textDecorationOverline
borderTopWidth	overflow	textDecorationUnderline
borderWidth	overflowX	textOverflow
bottom	overflowY	textTransform
clear	padding	textUnderlinePosition
clip	paddingBottom	top
color	paddingLeft	unicodeBidi
cursor	paddingRight	visibility
direction	paddingTop	whiteSpace
display	pixelBottom	width
filter	pixelHeight	wordSpacing
float	pixelLeft	wordWrap
font	pixelRight	writingMode
fontFamily	pixelTop	zIndex
fontSize	pixelWidth	zoom
fontStyle	posBottom	
fontVariant	posHeight	

link Objekt

backgroundPositionX	pixelLeft	posLeft
backgroundPositionY	pixelRight	posRight
behavior	pixelTop	posTop
Media	pixelWidth	posWidth
pixelBottom	posBottom	textAutospace
pixelHeight	posHeight	textUnderlinePosition

map Objekt

behavior

marquee Objekt

:first-letter	clip	pageBreakAfter
:first-line	color	pixelBottom
background	cursor	pixelHeight
backgroundAttachment	display	pixelLeft
backgroundColor	filter	pixelRight
backgroundImage	float	pixelTop
backgroundPosition	font	pixelWidth
backgroundPositionX	fontFamily	posBottom
backgroundPositionY	fontSize	posHeight
backgroundRepeat	fontStyle	position
behavior	fontVariant	posLeft
border	fontWeight	posRight
borderBottom	height	posTop
borderBottomColor	layoutFlow	posWidth
borderBottomStyle	layoutGrid	right
borderBottomWidth	layoutGridChar	styleFloat
borderColor	layoutGridLine	textAlign
borderLeft	layoutGridMode	textAutospace
borderLeftColor	layoutGridType	textDecoration
borderLeftStyle	left	textDecorationBlink
borderLeftWidth	letterSpacing	textDecorationLineThrough
borderRight	lineBreak	textDecorationNone
borderRightColor	lineHeight	textDecorationOverline
borderRightStyle	margin	textDecorationUnderline
borderRightWidth	marginBottom	textIndent
borderStyle	marginLeft	textJustify
borderTop	marginRight	textOverflow
borderTopColor	marginTop	textTransform
borderTopStyle	padding	textUnderlinePosition
borderTopWidth	paddingBottom	top
borderWidth	paddingLeft	unicodeBidi
bottom	paddingRight	visibility
clear	paddingTop	wordBreak



wordSpacing
writingMode

zIndex
zoom

namespace Objekt

keine

noframe Objekt

behavior
textAutospace

textUnderlinePosition
zoom

noscript Objekt

behavior
textAutospace

textUnderlinePosition
zoom

object Objekt

backgroundPositionX
backgroundPositionY
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear

clip
cursor
direction
display
float
height
layoutGrid
layoutGridMode
left
margin
marginBottom
marginLeft
marginRight
marginTop
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth

posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
scrollbar3dLightColor
scrollbarArrowColor
scrollbarBaseColor
scrollbarDarkShadowColor
scrollbarFaceColor
scrollbarHighlightColor
scrollbarShadowColor
scrollbarTrackColor
styleFloat
textAutospace
textUnderlinePosition
top
unicodeBidi
visibility
wordWrap
zoom



option ObjektbackgroundAttachment

backgroundColor
backgroundPositionX
backgroundPositionY
behavior
clear
color
direction
height
layoutFlow
layoutGrid

layoutGridMode
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight

posTop
posWidth
textAutospace
textUnderlinePosition
unicodeBidi
width
wordWrap
writingMode
zoom

popup Objekt

keine

runtimeStyle Objekt

alle Style-Eigenschaften des style Objektes (auch Pseudoklassen), die aber **nur** lesbar sind
Natürlich sind in `zeiger_auf_objekt.currentStyle.style_eigenschaft` nur die wirklich im Objekt implementierten Style-Eigenschaften referenziert.

screen Objekt

keine

script Objekt

backgroundPositionX
backgroundPositionY
behavior
layoutGrid
layoutGridMode
pixelBottom
pixelHeight

pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft

posRight
posTop
posWidth
textAutospace
textUnderlinePosition

span Objekt

accelerator
background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip
color

cursor
direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft

pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
verticalAlign
visibility
whiteSpace
width
wordSpacing
wordWrap
writingMode
zIndex
zoom

style Objekt

keine

styleSheet Objekt

cssText

media

textAutospace



table Objekt

accelerator	clip	pixelLeft
background	color	pixelRight
backgroundAttachment	cursor	pixelTop
backgroundColor	direction	pixelWidth
backgroundImage	display	posBottom
backgroundPosition	filter	posHeight
backgroundPositionX	float	position
backgroundPositionY	font	posLeft
backgroundRepeat	fontFamily	posRight
behavior	fontSize	posTop
border	fontStyle	posWidth
borderBottom	fontVariant	right
borderBottomColor	fontWeight	styleFloat
borderBottomStyle	height	tableLayout
borderBottomWidth	layoutGrid	textAlign
borderCollapse	layoutGridChar	textAutospace
borderColor	layoutGridLine	textDecoration
borderLeft	layoutGridMode	textDecorationBlink
borderLeftColor	layoutGridType	textDecorationLineThrough
borderLeftStyle	left	textDecorationNone
borderLeftWidth	letterSpacing	textDecorationOverline
borderRight	lineBreak	textDecorationUnderline
borderRightColor	lineHeight	textIndent
borderRightStyle	margin	textJustify
borderRightWidth	marginBottom	textTransform
borderStyle	marginLeft	textUnderlinePosition
borderTop	marginRight	top
borderTopColor	marginTop	unicodeBidi
borderTopStyle	padding	visibility
borderTopWidth	pageBreakAfter	wordBreak
borderWidth	pageBreakBefore	wordSpacing
bottom	pixelBottom	zIndex
clear	pixelHeight	zoom

table.caption Objekt

background	color	pixelHeight
backgroundAttachment	cursor	pixelLeft
backgroundColor	direction	pixelRight
backgroundImage	display	pixelTop
backgroundPosition	filter	pixelWidth
backgroundPositionX	font	posBottom
backgroundPositionY	fontFamily	posHeight
backgroundRepeat	fontSize	posLeft
behavior	fontStyle	posRight
border	fontVariant	posTop
borderBottom	fontWeight	posWidth
borderBottomColor	height	textAutospace
borderBottomStyle	layoutFlow	textDecoration
borderBottomWidth	layoutGrid	textDecorationBlink
borderColor	layoutGridMode	textDecorationLineThrough
borderLeft	letterSpacing	textDecorationNone
borderLeftColor	lineHeight	textDecorationOverline
borderLeftStyle	margin	textDecorationUnderline
borderLeftWidth	marginBottom	textOverflow
borderRight	marginLeft	textTransform
borderRightColor	marginRight	textUnderlinePosition
borderRightStyle	marginTop	unicodeBidi
borderRightWidth	padding	visibility
borderStyle	paddingBottom	width
borderTop	paddingLeft	wordSpacing
borderTopColor	paddingRight	writingMode
borderTopStyle	paddingTop	zIndex
borderTopWidth	pageBreakAfter	zoom
borderWidth	pageBreakBefore	
clear	pixelBottom	

table.col Objekt

background	backgroundPositionY	direction
backgroundAttachment	backgroundRepeat	display
backgroundColor	behavior	font
backgroundImage	clear	fontFamily
backgroundPosition	color	fontSize
backgroundPositionX	cursor	fontStyle



fontVariant
fontWeight
layoutGrid
layoutGridMode
letterSpacing
lineHeight
overflow
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom

pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight
posTop
posWidth
textAutospace
textDecoration

textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordSpacing
zIndex
zoom

table.colGroup Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
clear
color
cursor
direction
display
font
fontFamily
fontSize
fontStyle
fontVariant

fontWeight
layoutGrid
layoutGridMode
letterSpacing
lineHeight
overflow
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight

posLeft
posRight
posTop
posWidth
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textTransform
textUnderlinePosition
unicodeBidi
visibility
wordSpacing
zIndex
zoom

table.tBody Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
clear
color
cursor
direction
display
font
fontFamily
fontSize
fontStyle

fontVariant
fontWeight
layoutGrid
layoutGridMode
letterSpacing
lineHeight
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight

posTop
posWidth
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordSpacing
zIndex
zoom

table.tFoot Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
clear
color
cursor
direction
display
font
fontFamily
fontSize
fontStyle

fontVariant
fontWeight
layoutGrid
layoutGridMode
letterSpacing
lineHeight
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight

posTop
posWidth
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordSpacing
zIndex
zoom



table.tHead Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
clear
color
cursor
direction
display
font
fontFamily
fontSize
fontStyle

fontVariant
fontWeight
layoutGrid
layoutGridMode
letterSpacing
lineHeight
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
posLeft
posRight

posTop
posWidth
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordSpacing
zIndex
zoom

table.tr Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
clear
clip
color
cursor
direction
display
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height

layoutGrid
layoutGridChar
layoutGridLine
layoutGridMode
layoutGridType
letterSpacing
lineBreak
lineHeight
minHeight
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight

posTop
posWidth
textAlign
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textIndent
textJustify
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordBreak
wordSpacing
zIndex
zoom

table.tr.td Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth

borderWidth
clear
clip
color
cursor
direction
display
filter
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
hasLayout
height
layoutFlow
layoutGrid
layoutGridChar
layoutGridLine
layoutGridMode
layoutGridType
letterSpacing
lineBreak
lineHeight
margin
marginBottom
marginLeft

marginRight
marginTop
minHeight
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pageBreakBefore
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
textAlign
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone



textDecorationOverline
textDecorationUnderline
textIndent
textJustify
textTransform

textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordBreak

wordSpacing
wordWrap
writingMode
zIndex
zoom

table.tr.th Objekt

background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
clear
clip
color
cursor
direction

display
filter
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
hasLayout
height
layoutFlow
layoutGrid
layoutGridChar
layoutGridLine
layoutGridMode
layoutGridType
letterSpacing
lineBreak
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
minHeight
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pageBreakAfter
pageBreakBefore
pixelBottom
pixelHeight

pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
textAlign
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textIndent
textJustify
textTransform
textUnderlinePosition
unicodeBidi
verticalAlign
visibility
wordBreak
wordSpacing
wordWrap
writingMode
zIndex
zoom

textarea Objekt

accelerator
background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom

clear
clip
color
cursor
direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
imeMode
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding

paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom
pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
scrollbar3dLightColor
scrollbarArrowColor
scrollbarBaseColor
scrollbarDarkShadowColor
scrollbarFaceColor
scrollbarHighlightColor
scrollbarShadowColor
scrollbarTrackColor
styleFloat
textAlign
textAutospace
textDecoration
textDecorationBlink



textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform

textUnderlinePosition
top
unicodeBidi
visibility
width
wordSpacing

wordWrap
writingMode
zIndex
zoom

textrange Objekt

keine

textrectangle Objekt

keine

var Objekt

accelerator
background
backgroundAttachment
backgroundColor
backgroundImage
backgroundPosition
backgroundPositionX
backgroundPositionY
backgroundRepeat
behavior
border
borderBottom
borderBottomColor
borderBottomStyle
borderBottomWidth
borderColor
borderLeft
borderLeftColor
borderLeftStyle
borderLeftWidth
borderRight
borderRightColor
borderRightStyle
borderRightWidth
borderStyle
borderTop
borderTopColor
borderTopStyle
borderTopWidth
borderWidth
bottom
clear
clip

color
cursor
direction
display
filter
float
font
fontFamily
fontSize
fontStyle
fontVariant
fontWeight
height
layoutFlow
layoutGrid
layoutGridMode
left
letterSpacing
lineHeight
margin
marginBottom
marginLeft
marginRight
marginTop
overflow
overflowX
overflowY
padding
paddingBottom
paddingLeft
paddingRight
paddingTop
pixelBottom

pixelHeight
pixelLeft
pixelRight
pixelTop
pixelWidth
posBottom
posHeight
position
posLeft
posRight
posTop
posWidth
right
styleFloat
textAutospace
textDecoration
textDecorationBlink
textDecorationLineThrough
textDecorationNone
textDecorationOverline
textDecorationUnderline
textOverflow
textTransform
textUnderlinePosition
top
unicodeBidi
visibility
width
wordSpacing
wordWrap
writingMode
zIndex
zoom

window Objekt

keinef

xml Objekt

behavior

textAutospace

textUnderlinePosition

4.3.2.2.4.3.39.22. Verwaltung des Styles im Dokument**4.3.2.2.4.3.39.22.1. currentStyle Objekt des Internet Explorer**

referenziert **alle** zur Laufzeit **real im Element** vorhandenen **und zum Zeitpunkt aktuellen** Styles (CSS)

CSS kann für ein Element im Dokument definiert worden sein als

im Dokument global	z.B. per userdefinierte Style-Regel im HEAD
im Element als Inline-Styles	per STYLE-Attribut oder style Objekt
Pseudoklassen	

Script ab IE 5.x

Style-Arten und deren Verwendung bei Objekten: siehe Objekt style

Style-Eigenschaften sind **nur lesbar**

Hinweis: Objekt style repräsentiert **nur diejenigen** zur Laufzeit **real im Element** vorhandenen **und zum Zeitpunkt aktuellen** Styles (CSS), die als Inline-Style per STYLE-Attribut oder style Objekt erzeugt wurden, aber **keinen globalen** CSS und keine Pseudoklassen.

Styles können gelesen **und** geschrieben werden

Werte von CSS-Angaben lesen: es kann nur der Wert geliefert werden, der zugewiesen wurde bzw. Standard ist

Bsp: Zuweisung von "green"

lesen von "green" und **nicht** #00FF00

erst nach dem kompletten Laden des Dokumentes

unmittelbar nach dem Setzen einer CSS-Angabe nicht möglich



(Browser benötigt Zeit zur Verarbeitung)

Empfehlung: Funktionsaufruf je für Setzen und Lesen verschafft dem Browser Zeit

Syntax:

[var Wert =] object.currentStyle.style_eigenschaft

object Zeiger auf das Objekt mit CSS

nur lesen

Beispiel 1:

```

<HEAD>
<SCRIPT>
function BraunZuWeiss(ZeigerAufObjekt)
{
    if (ZeigerAufObjekt.currentStyle.backgroundColor == 'brown')
    { ZeigerAufObjekt.style.backgroundColor = 'white'; }
    else
    { ..... }
}
</SCRIPT>
</HEAD>
<BODY>
    <P STYLE="background-color: 'brown'" onclick=" BraunZuWeiss(this)">
</BODY>

```

Beispiel 2

```

<HEAD>
<STYLE>
    P { globaler_style: myvalue }
</STYLE>
<HEAD>
<BODY>
    <P ID="ID_P">
    <SCRIPT>
        alert(ID_P.currentStyle.globaler_style);
    </SCRIPT>
</BODY>

```

Beispiel 3

```

<BODY ID="ID_Body">
<TABLE BORDER=
    <TR>
        <TD WIDTH=1100 ID="ID_TD">
            text
        </TD>
    </TR>
</TABLE>
<SCRIPT>
    alert(
        "TD hat in currentStyle.width den Wert " + ID_TD.currentStyle.width
        + ".\nDie Breites des Fensters ist " + ID_Body.clientWidth + "px."
        + "\nDie Breite der Bildschirmauflösung ist " + screen.width + "px."
    );
</SCRIPT>
</BODY>

```

Beispiel 4:

```

<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect(0,100, "
                        + ID_select.options(ID_select.selectedIndex).value
                        + ",100)";

    if (ID_Image.currentStyle.clipBottom == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>

```



Eigenschaften:

alle Style-Eigenschaften des style Objektes (auch die Pseudoklassen) , die aber **nur** lesbar sind

.blockDirection	Umfluss um ein Objekt "ltr" Umfluss von links nach rechts "rtl" Umfluss von rechts nach links														
.clipBottom	untere Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt														
.clipLeft	linke Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt														
.clipRight	rechte Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt														
.clipTop	obere Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt														
.hasLayout	Objekt mit Style-Layout ab IE 5.5 Style-Layout wird erzeugt durch Kodierung einer Style-Eigenschaft (siehe style Objekt) oder durch Setzen der Eigenschaft .contentEditable auf true Folgende Objekte haben immer ein Style-Layout: BODY, IMG, INPUT, TABLE, TD Objekte mit <table> <tr> <td>.style.display</td><td>auf inline-block</td></tr> <tr> <td>.style.height</td><td></td></tr> <tr> <td>float</td><td>auf left oder right</td></tr> <tr> <td>.style.position</td><td>auf absolute</td></tr> <tr> <td>.style.width</td><td></td></tr> <tr> <td>.style.writingMode</td><td>auf tb-rl</td></tr> <tr> <td>.style.zoom</td><td></td></tr> </table>	.style.display	auf inline-block	.style.height		float	auf left oder right	.style.position	auf absolute	.style.width		.style.writingMode	auf tb-rl	.style.zoom	
.style.display	auf inline-block														
.style.height															
float	auf left oder right														
.style.position	auf absolute														
.style.width															
.style.writingMode	auf tb-rl														
.style.zoom															
	false Default. Objekt hat kein Layout														
	true Objekt hat Layout														
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound														

Methoden:

.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert		
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden <table> <tr> <td>expression()</td><td>oder setExpression()</td></tr> </table> zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)	expression()	oder setExpression()
expression()	oder setExpression()		
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert		
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <table> <tr> <td>objekt.style.eigenschaft.</td><td></td></tr> </table> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert	objekt.style.eigenschaft.	
objekt.style.eigenschaft.			

4.3.2.2.4.3.39.22.2. runtimeStyle Objekt des Internet Explorer

Objekt zum Überschreiben von CSS im currentStyle Objekt

Hinweise zum currentStyle Objekt:

aktueller CSS global (z.B. userdefinierte Style-Regel) **und** im Dokument (inline Styles oder STYLE-Attribut)

Script ab IE 5.x

Style-Arten und deren Verwendung bei Objekten: siehe Objekt style

Hinweis: Objekt style repräsentiert **nur** das STYLE-Attribut (inline Style) und keinen globalen CSS etc..

Werte von CSS-Angaben lesen: es kann nur der Wert geliefert werden, der zugewiesen wurde bzw. Standard ist

Bsp: Zuweisung von "green"

lesen von "green" und **nicht** #00FF00

erst nach dem kompletten Laden des Dokumentes

unmittelbar nach dem Setzen einer CSS-Angabe nicht möglich

(Browser benötigt Zeit)

Empfehlung: Funktionsaufruf je für Setzen und Lesen

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem

Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus



Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !!!

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

Syntax:

objekt.runtimeStyle.xxxxxx

mit xxxx als Style-Objekt-Eigenschaft

Beispiel 1:

```
<DIV onclick="this.runtimeStyle.cssText = 'color:red;background-color:blue;border:5px solid black;'">
    Das ist ein DIV. Klick fuer Style-Aenderung.
</DIV>
```

Beispiel 2:

```
<SCRIPT>
    function FarbeAendern(FarbeNameAlsKette)
    {
        if (ID_DIV.runtimeStyle.backgroundColor != ID_DIV.style.backgroundColor)
        { ID_DIV.runtimeStyle.backgroundColor = FarbeNameAlsKette; }

        alert(    ID_DIV.style.backgroundColor + "\n"
                +    ID_DIV.currentStyle.backgroundColor + "\n"
                +    ID_DIV.runtimeStyle.backgroundColor
                );
    }
</SCRIPT>
<DIV ID= "ID_DIV">
    Das ist ein DIV
</DIV>
<INPUT TYPE = "button" VALUE = "FarbeAendern" onclick="FarbeAendern('blue')">
```

Eigenschaften:

alle Eigenschaften des style Objektes

.onOffBehavior deprecated ab IE 5.x
Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound

Methoden:

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
DOM nicht geändert

.getExpression() Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern
Style-Eigenschaft ist per Methoden
expression() oder setExpression()
zu definieren
DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout
(nach dem eventuellen expliziten Dokument-Refresh)

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
DOM wird geändert

.removeExpression() Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient.
Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
DOM wird nicht geändert

.setAttribute() Wert von vorhandenem Attribut setzen
wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
DOM wird nur bei Erzeugung geändert

.setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient
Ausdruck nur als Script kodierbar
DOM wird nicht geändert

4.3.2.2.4.3.39.23. Standard-Behavior (Verhaltensweisen) von Objekten des Internet Explorer (Auswahl)

Behavior sind erst ab IE 5.x möglich, z.T. erst ab IE 5.5 bzw. IE 6.x..

Das Objekt document.namespaces ist erst ab IE 5.5 implementiert.

Eine Erweiterung der Verhaltensweisen eines Elementes (Objektes) kann auch über extern definierte Anweisungen erfolgen. Diese Anweisungen liegen nicht im Dokument sondern in einer HTML Component (HTC) Datei vom Typ *.htc. Zur Wirksamkeit der Anweisungen muss die Datei in das Dokument geladen werden. Die Verhaltensweise ist also erst nach dem kompletten Laden - inklusive der *.htc-Datei - nutzbar (window.onload Event muss ausgelöst sein).



Eine *.htc-Datei muss aus Anweisungen bestehen, die der Browser interpretieren und ausführen kann, um damit das Verhalten von Elementen zu realisieren. Das gilt vorallem für ein privates Behavior. Siehe unbedingt document.namespace Objekt.
Zu Aufbau und Funktionsweise der *.htc-Datei siehe Objekt element.

Ein Behavior kann per STYLE-Attribut, Script oder XML implementiert werden.

In HTML und XML muss ein ID-Attribut kodiert werden !

Der Internet Explorer besitzt Standard-Behavior, die nachfolgend beschrieben werden.

Es ist nicht nötig, eine *.htc-Datei einzubinden, da die StandardBehavior im Browser selbst liegen.

Diese Verhaltensweisen haben Objektcharakter und besitzen z.T. Eigenschaften wie Methoden.

Aus Übersichtsgründen werden Behavior nicht als Objekte beschrieben, da sie aus Sicht von Javascript direkt mit dem style-Objekt in Verbindung stehen.

Beispiele für diverse Kodierungsformen der Standard-Behavior, außer für .style.time2 Behavior: siehe dort

Beispiel für Einbindung eines XML-Namensraumes:

```
<HTML XMLNS:IE>
<HEAD>
<STYLE>
    @media all {IE\;clientCaps {behavior:url(#default#clientCaps)}}
</STYLE>
<SCRIPT>
    function window.onload()
    {
        ID_Pre.innerText =      "availHeight      = " + ID_ClientCaps.availHeight
                                + "\n" + "availWidth    = " + ID_ClientCaps.availWidth
                                + "\n" + "bufferDepth   = " + ID_ClientCaps.bufferDepth
                                + "\n" + "colorDepth    = " + ID_ClientCaps.colorDepth
                                + "\n" + "connectionType = " + ID_ClientCaps.connectionType
                                + "\n" + "cookieEnabled = " + ID_ClientCaps.cookieEnabled
                                + "\n" + "cpuClass      = " + ID_ClientCaps.cpuClass
                                + "\n" + "height       = " + ID_ClientCaps.height
                                + "\n" + "javaEnabled  = " + ID_ClientCaps.javaEnabled
                                + "\n" + "platform    = " + ID_ClientCaps.platform
                                + "\n" + "systemLanguage = " + ID_ClientCaps.systemLanguage
                                + "\n" + "userLanguage = " + ID_ClientCaps.userLanguage
                                + "\n" + "width       = " + ID_ClientCaps.width
                                + "\n" ;
    }
</SCRIPT>
</HEAD>
<BODY>
    <IE:clientCaps ID="ID_ClientCaps" >
    <PRE ID="ID_Pre"></PRE>
</BODY>
</HTML>
```

Beispiel für Behavior-Deklaration direkt im STYLE-Attribut:

```
<HEAD>
<SCRIPT>
<!--
    function window.onload() // Achtung: überschreibt Standard-onload-Routine !!
    {
        var Kette = ID_ClientCaps.getComponentVersion(
            "{89820200-ECBD-11CF-8B85-00AA005B4383}",
            "componentid"
        );

        ID_Div.innerHTML =  "<FONT SIZE=4>Internet Explorer Version = "
                            +      Kette
                            + ".</FONT>";
    }
-->
</SCRIPT>
</HEAD>
<BODY  BGCOLOR="#FFFFFF"
      STYLE="behavior:url(#default#clientCaps)" ID="ID_ClientCaps"
>
    <DIV ID="ID_Div"></DIV>
</BODY>
```

Beispiel für XML-Namensraum und STYLE-Kodierung:



```

<HTML XMLNS:IE>
<HEAD>
<SCRIPT>
    function NachDemDownload(DownLoadedFileContent)
    {alert (DownLoadedFileContent);}
</SCRIPT>
</HEAD>
<BODY>
    <IE:Download ID="ID_IETag" STYLE="behavior:url(#default#download)" >
    <P>
    Click
    <A HREF="javascript:ID_IETag.startDownload('download.htm', NachDemDownload)">
    hier
    </A>
    zum Start des Downloads dieser Seite.
</BODY>
</HTML>

```

Beispiel für Behavior-Deklaration durch Zuweisung auf das STYLE-Attribut:

```

<HEAD>
<SCRIPT>
    function window.onload()
    {
        ID_ClientCaps.style.behavior = "url(#default#clientCaps)";

        var DataBindingVerfuegbar = ID_ClientCaps.isComponentInstalled(
            "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
            "componentid"
        );

        if (!DataBindingVerfuegbar)
        {
            ID_ClientCaps.addComponentRequest(
                "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
                "componentid"
            );

            DataBindingVerfuegbar = ID_ClientCaps.doComponentRequest();

            alert(DataBindingVerfuegbar);
        }
    }
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF" ID="ID_ClientCaps">
</BODY>

```

Beispiel für Kodierung per Klasse im HEAD:

```

<HEAD>
<STYLE>
    .FolderKlasse { behavior:url(#default#httpFolder);}
</STYLE>
<SCRIPT>
    function Anzeigen()
    {
        var Ordner=location.href.substring(0,location.href.lastIndexOf("/"));
        ID_Span.navigate(Ordner);
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" CLASS = "FolderKlasse" onclick = "Anzeigen()">
    </SPAN>
</BODY>

```

4.3.2.2.4.3.39.23.1. .style.clientCaps Behavior des Internet Explorer

verwaltet Informationen vom Internet Explorer unterstützten Standard-Verhaltensarten der Elemente ab IE 5.x

siehe auch document.namespace Objekt

Syntax:

XML	<Prefix: CustomTag ID=kette STYLE="behavior:url('#default#clientCaps')">
HTML	<ELEMENT STYLE="behavior:url('#default#clientCaps')" ID= kette >
Scripting	object.style.behavior = "url('#default#clientCaps')"
	object.addBehavior ("#default#clientCaps")



Hinweis zur XML-Kodierung:
Anstelle der Tagbegrenzung >

sollte ein eigenständiges Endetag kodiert werden, also

```
.....>
....
</.....>
```

Prefix	laut XML-Namensraum
CustomTag	User-definierter Tag
kette	String mit dem ID
	muss kodiert werden

Beispiel für Eigenschaften:

```
<HTML XMLNS:IE>
<HEAD>
<STYLE>
    @media all {IE\:clientCaps {behavior:url(#default#clientCaps)}}
</STYLE>
<SCRIPT>
    function window.onload()
    {
        ID_Pre.innerText =
            "availHeight          = " + ID_ClientCaps.availHeight
            + "\n" + "availWidth    = " + ID_ClientCaps.availWidth
            + "\n" + "bufferDepth   = " + ID_ClientCaps.bufferDepth
            + "\n" + "colorDepth    = " + ID_ClientCaps.colorDepth
            + "\n" + "connectionType = " + ID_ClientCaps.connectionType
            + "\n" + "cookieEnabled = " + ID_ClientCaps.cookieEnabled
            + "\n" + "cpuClass      = " + ID_ClientCaps.cpuClass
            + "\n" + "height        = " + ID_ClientCaps.height
            + "\n" + "javaEnabled   = " + ID_ClientCaps.javaEnabled
            + "\n" + "platform      = " + ID_ClientCaps.platform
            + "\n" + "systemLanguage = " + ID_ClientCaps.systemLanguage
            + "\n" + "userLanguage  = " + ID_ClientCaps.userLanguage
            + "\n" + "width         = " + ID_ClientCaps.width
            + "\n";
    }
</SCRIPT>
</HEAD>
<BODY>
    <IE:clientCaps ID="ID_ClientCaps" >
    <PRE ID="ID_Pre"></PRE>
</BODY>
</HTML>
```

Beispiel 1 für Methoden:

```
<HEAD>
<SCRIPT>
<!--
    function window.onload() // Achtung: überschreibt Standard-onload-Routine !!
    {
        var Kette = ID_ClientCaps.getComponentVersion(
            "{89820200-ECBD-11CF-8B85-00AA005B4383}",
            "componentid"
        );

        ID_Div.innerHTML = " <FONT SIZE=4>Internet Explorer Version = "
            + Kette
            + ".</FONT>";
    }
-->
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF"
    STYLE="behavior:url(#default#clientCaps)" ID="ID_ClientCaps"
>
    <DIV ID="ID_Div"></DIV>
</BODY>
```

Beispiel 2 für Methoden:



```

<HEAD>
<SCRIPT>
    function window.onload()
    {
        ID_ClientCaps.style.behavior = "url(#default#clientCaps)";

        var DataBindingVerfuegbar = ID_ClientCaps.isComponentInstalled(
            "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
            "componentid"
        );

        if (!DataBindingVerfuegbar)
        {
            ID_ClientCaps.addComponentRequest(
                "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
                "componentid"
            );

            DataBindingVerfuegbar = ID_ClientCaps.doComponentRequest();

            alert(DataBindingVerfuegbar);
        }
    }
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF" ID="ID_ClientCaps">
</BODY>

```

Eigenschaften:

.availHeight	verfügbare Höhe des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar Behavior .style.clientCaps
.availWidth	verfügbare Breite des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar Behavior .style.clientCaps
.bufferDepth	Anzahl der Bits pro Pixel für eine Farbe UND Verwendung des off-screen bitmap buffer Behavior .style.clientCaps
.colorDepth	Anzahl der Bits pro Pixel für eine Farbe UND Verwendung destination device or buffer wird durch den Wert der Eigenschaft bufferDepth überschrieben, wenn .bufferDepth mit Wert > 0 Behavior .style.clientCaps
.connectionType	Typ der genutzten Verbindung bzw. Offline-Status der Verbindung Behavior .style.clientCaps
.cookieEnabled	Cookienutzbarkeit im Browser Behavior .style.clientCaps
.cpuClass	CPU-Hersteller Behavior .style.clientCaps
.height	Auflösung des Bildschirms in Höhe, also Anzahl der vertikalen Pixel Behavior .style.clientCaps
.javaEnabled	Verfügbarkeit der Microsoft Virtual Machine (Microsoft VM) für Java Achtung: Aufgrund eines Rechtsstreites mit dem Unternehmen Sun entwickelt Microsoft ab IE 6.x die MS VM nicht mehr weiter, hält die VM aber noch per Download verfügbar. Für eine aktuellere Version muss der User selbständig beim Unternehmen Sun eine Lizenz für Java erwerben und die zugehörige Software installieren (inklusive Updates). Es reicht dabei die Run-Time-Version der Virtuellen Java Maschine von Sun. Ob allerdings Sun-Java-Standards in Microsoft-Produkten implementiert werden, ist nicht gesichert.
.platform	Behavior .style.clientCaps Windows-Betriebssystem Behavior .style.clientCaps
.systemLanguage	Standardsprache des Betriebssystems Behavior .style.clientCaps
.userLanguage	Sprache des Betriebssystems laut UserEinstellung Behavior .style.clientCaps
.width	Auflösung des Bildschirms in Breite, also Anzahl der horizontalen Pixel Behavior .style.clientCaps

Methoden:

Nachfolgende Methoden behandeln die im IE per Active-X (Active Setup) installierbaren vordefinierten Komponenten. Der Umfang der Komponenten ist abhängig von der IE-Version. Das Installieren der Komponente ist das Binden von deren Daten in den IE (Active Setup). Die Komponenten werden dabei in einem Puffer (Queue) gehalten. Dieser Puffer wird durch den Download der Komponente gefüllt. Es kann nur eine gepufferte Komponente installiert werden.

ID der Komponenten werden in Gross- und Kleinschreibung unterschieden.

Ein ID ist z.B. der String "{9381D8F2-0288-11D0-9501-00AA00B911A5}"

Typ der Komponente ist immer "componentid", wobei Gross-Klein egal ist

Beispiel 1 für Methoden:



```

<BODY BGCOLOR="#FFFFFF"
  STYLE="behavior:url(#default#clientCaps)" ID="ID_ClientCaps"
>
  <DIV ID="ID_Div"></DIV>
  <SCRIPT>
  <!--
    function window.onload()
    {
      var Kette = ID_ClientCaps.getComponentVersion(
        "{89820200-ECBD-11CF-8B85-00AA005B4383}",
        "componentid"
      );

      ID_Div.innerHTML = "<FONT SIZE=4>Internet Explorer Version = "
        + Kette
        + "</FONT>";
    }
  -->
  </SCRIPT>
</BODY>

```

Beispiel 2 für Methoden:

```

<HEAD>
<SCRIPT>
  function window.onload()
  {
    ID_ClientCaps.style.behavior = "url(#default#clientCaps)";

    var DataBindingVerfuegbar = ID_ClientCaps.isComponentInstalled(
      "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
      "componentid"
    );

    if (!DataBindingVerfuegbar)
    {
      ID_ClientCaps.addComponentRequest(
        "{9381D8F2-0288-11D0-9501-00AA00B911A5}",
        "componentid"
      );

      DataBindingVerfuegbar = ID_ClientCaps.doComponentRequest();

      alert(DataBindingVerfuegbar);
    }
  }
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF" ID="ID_ClientCaps">
</BODY>

```

.addComponentRequest()	Komponente zum Download anfordern und nach dem Download installieren Behavior .style.clientCaps
.clearComponentRequest()	Puffer der per .addComponentRequest() angeforderten Downloads löschen Behavior .style.clientCaps
.compareVersions()	2 Versionen einer Komponente vergleichen Bsp. für Version "5,0,18,1024"
.doComponentRequest()	Start des Download aller per .addComponentRequest() angeforderten Komponenten laut Puffer Behavior .style.clientCaps
.getComponentVersion()	Version einer Komponente Behavior .style.clientCaps
.isComponentInstalled()	Verfügbarkeit der Komponente (gedownload UND installiert UND aktuelle Versionsnummer ist mindestens die minimale Versionsnummer) Behavior .style.clientCaps

4.3.2.2.4.3.39.23.2. .style.download Behavior des Internet Explorer

Download einer Datei und Aufruf einer freiprogrammierbaren Funktion direkt nach dem kompletten Ende des Downloads
Behavior erst wirksam nach dem kompletten Laden des Dokumentes (window.onload Event muss ausgelöst worden sein)
Kodierung nur innerhalb derselben Domain (downzuladende Datei muss in selber Domain liegen)

ab IE 5.x

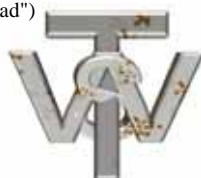
siehe auch document.namespace Objekt

Syntax:

```

XML    <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#download')" >
HTML   <ELEMENT STYLE="behavior:url('#default#download')" ID=kette>
Scripting object.style.behavior = "url('#default#download')"
        object.addBehavior("#default#download")

```



Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

.....>
....
</.....>

Prefix laut XML-Namensraum
CustomTag User-definierter Tag
kette String mit dem ID
muss kodiert werden

Beispiel:

```
<HTML XMLNS:IE>
<HEAD>
<SCRIPT>
    function NachDemDownload(DownLoadedFileContent)
    { alert (DownLoadedFileContent); }
</SCRIPT>
</HEAD>
<BODY>
    <IE:Download ID="ID_IETag" STYLE="behavior:url(#default#download)" >
    <P>
    Click
    <A HREF="javascript:ID_IETag.startDownload('download.htm', NachDemDownload)">
    hier
    </A>
    zum Start des Downloads dieser Seite.
</BODY>
</HTML>
```

Eigenschaften:

keine

Methoden:

.startDownload() Start des Download

4.3.2.2.4.3.39.23.3. *.style.homePage Behavior des Internet Explorer*

enthält die Informationen nur für Homepages innerhalb einer gemeinsamen Domain

ab IE 5.x

siehe auch document.namespace Objekt

Syntax:

XML	< Prefix: CustomTag ID= kette STYLE="behavior:url('#default#homePage')">
HTML	<ELEMENT STYLE="behavior:url('#default#homePage')" ID= kette >
Scripting	object.style.behavior = "url('#default#homePage')" object.addBehavior ("#default#homePage")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

.....>
....
</.....>

Prefix laut XML-Namensraum
CustomTag User-definierter Tag
kette String mit dem ID
muss kodiert werden

Beispiel

```
<HTML XMLNS:IE>
<HEAD>
    <STYLE>
        @media all { IE\;homePage { behavior:url(#default#homepage)} }
    </STYLE>
<SCRIPT>
    function HomepageErmitteln()
    {
        alert(ID_BenutzerTag.isHomePage(ID_Input.value));
        event.returnValue = false;
    }

    function HomepageSetzen()
    {
```




```

        ID_BenutzerTag.setHomePage(ID_Input.value);
        event.returnValue = false;
    }

    function HomepageAnspringen()
    {
        ID_BenutzerTag.navigateHomePage();
        event.returnValue=false;
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <IE:homePage ID="ID_BenutzerTag" >
    <INPUT TYPE=text ID="ID_Input" VALUE="http://www.test.de" >
    <INPUT TYPE=button VALUE="Homepage ermitteln" onclick="HomepageErmitteln()">
    <INPUT TYPE=button VALUE="Homepage setzen" onclick="HomepageSetzen()">
    <INPUT TYPE=button VALUE=" Homepage anspringen" onclick="HomepageAnspringen()">
</BODY>
</HTML>

```

Beispiel für Bookmark (Favoriten) setzen

```

<a href="#" onClick="this.style.behavior='url(#default#homepage)';
this.setHomePage('http://www.test.de');">Machen Sie test.de zu Ihrer Startseite</a>

```

Eigenschaften:

keine

Methoden:

.isHomePage()	Lage der Homepage auf einer Domain
.navigateHomePage()	Homepage anspringen im Browser
.setHomePage()	Homepage-Lade-Dialogbox aufrufen

4.3.2.2.4.3.39.23.4. .style.httpFolder Behavior des Internet Explorer

Darstellung eines HTTP-Verzeichnis im Fenster

ab IE 5.x

siehe auch document.namespace Objekt

Syntax:

XML	<Prefix: CustomTag ID=kette STYLE="behavior:url('#default#httpFolder')" >
HTML	<ELEMENT STYLE="behavior:url('#default#httpFolder')" ID=kette>
Scripting	object.style.behavior = "url('#default#httpFolder')"
	object.addBehavior ("#default#httpFolder")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

```

.....>
....
</.....>

```

Prefix	laut XML-Namensraum
CustomTag	User-definierter Tag
kette	String mit dem ID
	muss kodiert werden

Beispiel

```

<HEAD>
<STYLE>
    .FolderKlasse { behavior:url('#default#httpFolder');}
</STYLE>
<SCRIPT>
    function Anzeigen()
    {
        var Ordner=location.href.substring(0,location.href.lastIndexOf("/"));
        ID_Span.navigate(Ordner);
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" CLASS = "FolderKlasse" onclick = "Anzeigen()">
</SPAN>
</BODY>

```

Eigenschaften:

keine

Methoden:

.navigate()	HTTP-Verzeichnis im aktuellen Fenster anzeigen
-------------	--



.navigateFrame() HTTP-Verzeichnis im Fenster nach Wahl anzeigen

4.3.2.2.4.3.39.23.5. *.style.mediaBar Behavior des Internet Explorer*

Basis-User-Interface für Media-Wiedergabe im Browser-Fenster der Media Bar (Medien-Fenster) mit diversen Funktionalitäten z.B.

Steuerung der Wiedergabe

Navigation per HTML z.B. Link

Playersoftware (Interface) Windows Media Player als Media Bar Player

laden von HTML in das Medienfenster z.B. per Link, der auf das Ziel "_media" verweisen muss, also das Medien-Fenster

Beispiel für HTML-Link

```
<A HREF="http://www.test.de/test.asf" TARGET="_media"></A>
<A HREF="http://www.test.de/start.htm" TARGET="_media"></A>
```

ab IE 6.x nur unter Windows 32Bit

öffnen der Media Bar durch

Userklick auf Link im Browserfenster

Userklick auf Medien-Button in der Menüleiste des IE

Usereingabe in der Adresszeile einer Url von einer Media-Datei

Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen.

Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die aktuelle Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei.

Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt.

Wird die Media-Datei nicht gefunden oder ist der Mime-Typ der Media-Datei unbekannt, dann wird eine Standard-Seite in das Medien-Fenster geladen.

Wird während der HTML-Navigation ein Ziel nicht geladen (weil z.B. nicht existent), dann wird eine Standard-Seite in das Medien-Fenster geladen.

Scriptsteuerung des Behavior erst möglich, wenn window.onload Event erzeugt wurde.

Das Dokument, das die Behavior-Deklaration enthält, muss in die Media Bar geladen werden, damit das Behavior nutzbar ist.

Bsp.:

start.htm enthält die Behavior-Deklaration

Syntax:

```
XML <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#mediaBar')">
HTML <ELEMENT STYLE="behavior:url('#default#mediaBar')" ID=kette>
Scripting object.style.behavior = "url('#default#mediaBar')";
        object.addBehavior ("#default#mediaBar")
```

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

```
.....>
....
</.....>
```

Prefix laut XML-Namensraum
CustomTag User-definierter Tag
kette String mit dem ID
muss kodiert werden

Beispiel:

```
<DIV ID="ID_Div"
STYLE="behavior:url('#default#mediaBar')">
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
ID_Div.enabled = false;
"
>
```

Eigenschaften:

.currentItem

Zeiger auf aktuellen Media-Eintrag (Objekt MediaItem)

Eintrag stammt aus der Playliste, wenn diese existent ist, und ist der aktuelle Eintrag der Playliste



.disabledUI	siehe Behavior .style.mediaBar Sichtbarkeit des User-Interfaces der Media Bar
.enabled	siehe Behavior .style.mediaBar Sichtbarkeit des Media Bar Player Media Bar Player ist der Windows Media Player
.hasNextItem	siehe Behavior .style.mediaBar aktueller Eintrag in der Playliste (Objekt PlaylistInfo) hat einen Nachfolger-Eintrag
.nextItem	siehe Behavior .style.mediaBar Zeiger auf nächsten Eintrag in der Playliste (Objekt PlaylistInfo)
.openState	siehe Behavior .style.mediaBar Status des Media Bar Player bezüglich Playlist, Codec, Lizenz und Individualisierung Media-Datei kann haben Codec (Ländernummer) Lizenz bei käuflichem Erwerb durch den User Individualisierung auf den User
.playlistInfo	Media Bar Player ist der Windows Media Player Zeiger auf die Playliste (Objekt PlaylistInfo) siehe Behavior .style.mediaBar
.playState	Wiedergabe-Status des Media Bar Player (Wiedergabe der Media-Datei) Status wird verändert durch Aktionen des Users mit dem Player Media Bar Player ist der Windows Media Player Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen. Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die
aktuelle	Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei. siehe Methoden .playNext() .playURL() .stop() siehe Behavior .style.mediaBar
Methoden:	
.playNext()	Wiedergabe des nächsten Eintrages in der Playliste (Objekt PlaylistInfo) starten verändert Eigenschaft .playState Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen. Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die
aktuelle	Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei. Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt. Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar: .getItemInfo() .getAttributeName() Eigenschaften nutzbar: .attributeCount Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen. siehe Methoden .playURL() .stop() und Eigenschaft .playState siehe Behavior .style.mediaBar
.playURL()	Laden einer Media-Datei in die Media Bar und wenn Laden erfolgreich, so Wiedergabe der Media-Daten Achtung: Wird die Media-Datei nicht gefunden oder ist der Mime-Typ der Media-Datei unbekannt, dann wird eine Standard-Seite in das Medien-Fenster geladen. verändert Eigenschaft .playState Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen. Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die
aktuelle	Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei. Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt. Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar: .getItemInfo() .getAttributeName() Eigenschaften nutzbar: .attributeCount Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen. siehe Methoden .playNext() .stop() und Eigenschaft .playState siehe Behavior .style.mediaBar
.stop()	aktive Wiedergabe stoppen, also beenden verändert Eigenschaft .playState Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen. Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die
aktuelle	Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei. Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt.



Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende

Methoden nutzbar:

.getItemInfo()
.getAttributeName()

Eigenschaften nutzbar:

.attributeCount

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

Achtung: Wird die Methode .stop() innerhalb von 10 Sekunden nach dem ERSTEN Aufruf erneut aufgerufen, dann wird eine Standard-Seite in das Medien-Fenster geladen.

siehe Methoden .playURL() .playNext() und Eigenschaft .playState

siehe Behavior .style.mediaBar

Events:

onhide

erzeugt wenn der Media Bar Player gerade unsichtbar gemacht wird (versteckt wird)

siehe Eigenschaft .enabled

entspricht dem Schliessen des Media Bar Player durch den User

Media Bar Player ist der Windows Media Player

siehe Behavior .style.mediaBar

onopenstatechange

erzeugt, wenn Media Bar Player den Status bezüglich Playliste, Codec, Lizenz und

Individualisierung ändert

siehe Eigenschaft .openState

Media Bar Player ist der Windows Media Player

siehe Behavior .style.mediaBar

onplaystatechange

erzeugt, wenn Media Bar Player den Status bezüglich Wiedergabe ändert

siehe Eigenschaft .playState

Media Bar Player ist der Windows Media Player

siehe Behavior .style.mediaBar

onshow

erzeugt wenn der Media Bar Player gerade sichtbar gemacht wird (nicht versteckt wird)

siehe Eigenschaft .enabled

entspricht dem Öffnen des Media Bar Player durch den User

Media Bar Player ist der Windows Media Player

siehe Behavior .style.mediaBar

MediaItem Objekt des Internet Explorer

Repräsentiert für die Media Bar den aktuellen Eintrag in der Playliste (Objekt PlaylistInfo), falls Playliste existent ist

die aktuelle Media-Datei, wenn keine Playliste existent ist

Syntax:

zeiger_auf_mediaBar.MediaItem.eigenschaft

zeiger_auf_mediaBar.MediaItem.methode

zeiger_auf_mediaBar

laut ID-Attribut

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende

Methoden nutzbar:

.getItemInfo()
.getAttributeName()

Eigenschaften nutzbar:

.attributeCount

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

Eigenschaften:

.attributeCount

Anzahl der mit dem MediaItem Objekt verbundenen Attribute liefern

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende

Methoden nutzbar:

.getItemInfo()
.getAttributeName()

Eigenschaften nutzbar:

.attributeCount

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

siehe Behavior .style.mediaBar

.duration

Dauer des MediaItem Objektes in Sekunden

siehe Behavior .style.mediaBar

.name

Name des MediaItem Objektes

siehe Behavior .style.mediaBar

.sourceURL

Url der Media-Datei des MediaItem Objektes

siehe Behavior .style.mediaBar

Methoden:

.getAttributeName()

Name des mit dem MediaItem Objekt verbundenen Attributes liefern

z.B. zur Feststellung, welche Attribute eine Advanced Stream Redirector (ASX)-Datei hat wie abstract oder author oder copyright

Beispiel für Aufbau eines Eintrages in einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
```

```
<entry>
```

```
<title>Testtitel</title>
```

```
<author>Testautor</author>
```



```

<copyright>Test 2002</copyright>
<abstract>WAV Datei</abstract>
<ref href=""></ref>
<banner href = "Testbild.gif" >
    <moreinfo href = "Test.doc"></moreinfo>
    <abstract>besuche www.test.de</abstract>
</banner>
</entry>
</ASX>

```

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

```

.getItemInfo()
.getAttributeName()

```

Eigenschaften nutzbar:

```

.attributeCount

```

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

siehe Behavior .style.mediaBar

.getItemInfo()

Wert des mit dem MediaItem Objekt verbundenen Attributes liefern

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

```

.getItemInfo()
.getAttributeName()

```

Eigenschaften nutzbar:

```

.attributeCount

```

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

siehe Behavior .style.mediaBar

PlaylistInfo Objekt des Internet Explorer:

Repräsentiert die Playlist der Media Bar.

Syntax:

```

zeiger_auf_mediaBar.PlaylistInfo.eigenschaft
zeiger_auf_mediaBar.PlaylistInfo.methode

```

zeiger_auf_mediaBar laut ID-Attribut

Eigenschaften:

.attributeCount

Anzahl der mit dem MediaItem Objekt verbundenen Attribute liefern

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

```

.getItemInfo()
.getAttributeName()

```

Eigenschaften nutzbar:

```

.attributeCount

```

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.

siehe Behavior .style.mediaBar

.name

Name des MediaItem Objektes

siehe Behavior .style.mediaBar

Methoden:

.getAttributeName()

Name des mit dem MediaItem Objekt verbundenen Attributes liefern

z.B. zur Feststellung, welche Attribute eine Advanced Stream Redirector (ASX)-Datei hat wie abstract oder author oder copyright

Beispiel für Aufbau eines Eintrages in einer ASX-Datei:

```

<ASX Version="1.0" PreviewMode="No" >
<entry>
    <title>Testitel</title>
    <author>Testautor</author>
    <copyright>Test 2002</copyright>
    <abstract>WAV Datei</abstract>
    <ref href=""></ref>
    <banner href = "Testbild.gif" >
        <moreinfo href = "Test.doc"></moreinfo>
        <abstract>besuche www.test.de</abstract>
    </banner>
</entry>
</ASX>

```

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

```

.getItemInfo()
.getAttributeName()

```

Eigenschaften nutzbar:

```

.attributeCount

```



Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.
 siehe Behavior .style.mediaBar
 Wert des mit dem MediaItem Objekt verbundenen Attributes liefern
 Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:
 .getItemInfo()
 .getAttributeName()
 Eigenschaften nutzbar:
 .attributeCount

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.
 siehe Behavior .style.mediaBar

4.3.2.2.4.3.39.23.6. .style.saveFavorite Behavior des Internet Explorer

frei definierbare Daten aus dem Dokument sichern bzw. laden:

sichern, wenn **User** das Dokument mit seiner Url als Bookmark zu den Favoriten hinzufügt
 oder User (auch ohne Bookmarksetzung) das Dokument verlässt
 laden, wenn **User** das Dokument aus der Favoritenliste wieder anwählt

ab IE 5.x

siehe auch document.namespace Objekt

Es werden folgende Events vom Objekt **mit** dem saveFavorit-Behavior ausgelöst:

onload	wenn Dokument per Favorit angewählt und somit geladen wird
onsave	wenn Dokument als Bookmark in die Favoritenliste eingetragen wird
oder	wenn User das Dokument verlässt (ohne Bookmarksetzung)

Gespeichert wird immer permanent, also auch über mehrere Online-Sitzungen, also
 in den browserinternen Datenbereich
 und zugleich physisch auf der Festplatte des Users als INI-Datei.
 Pro Objekt wird dazu ein objekt eigener User-Data-Store verwendet.

Bsp. für Aufbau der url-Datei in der Favoritenliste:

```
[DEFAULT]
BASEURL=http://www.test.de/
[InternetShortcut]
URL=http://www.test.de/
Modified=009ECB4522B6C10148
IconFile=http://www.test.de/favicon.ico
IconIndex=1
```

Der Eintrag Modified=009ECB4522B6C10148 wird vom Browser festgelegt, so bald die Bookmark erzeugt wird, und ist somit Bookmark-spezifisch.

Syntax:

XML	< Prefix: CustomTag ID= kette STYLE="behavior:url('#default#saveFavorite')">
HTML	<ELEMENT STYLE="behavior:url('#default#saveFavorite')" ID= kette >
Scripting	object.style.behavior = "url('#default#saveFavorite')" object.addBehavior("#default#saveFavorite")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

```
.....>
....
</....>
```

Prefix	laut XML-Namensraum
CustomTag	User-definierter Tag
kette	String mit dem ID muss kodiert werden

Beispiel

```
<HTML>
<HEAD>
  <STYLE>
    .FavoriteKlasse {behavior:url('#default#saveFavorite');}
  </STYLE>
  <SCRIPT>
    function InterneDateSpeichern()
    {
      // Textwert des Input als neues Attribut erzeugen und speichern
      ID_Input.setAttribute("InterneDateAlsAttribut", ID_Input.value);
    }
  </SCRIPT>
```



```

function InterneDateLaden()
{
    // Textwert des Input laden
    ID_Input.value= ID_Input.getAttribute("InterneDateAlsAttribut");
}
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE=text
        ID="ID_Input"
        CLASS= FavoriteKlasse
        onsave="InterneDateSpeichern()"
        onload="InterneDateLaden()"
    >
</BODY>
</HTML>

```

Eigenschaften:

.XMLDocument Referenz auf XML-Dokument (XML-DOM)

Methoden:

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
DOM nicht geändert

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
DOM wird geändert

.setAttribute() Wert von vorhandenem Attribut setzen
wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
DOM wird nur bei Erzeugung geändert

4.3.2.2.4.3.39.23.7. .style.saveHistory Behavior des Internet Explorer

frei definierbare Daten aus dem Dokument sichern bzw. laden:

sichern, wenn **User** das Dokument verlässt

laden, wenn **User** das Dokument wieder anwählt per Zurück- oder Vorwärts-Button im Browser-Menü

ab IE 5.x

siehe auch document.namespace Objekt

Es werden folgende Events vom Objekt **mit** dem saveHistory-Behavior ausgelöst:

onload	wenn Dokument geladen wird per Zurück- oder Vorwärts-Button im Browser-Menü
onsave	wenn Dokument verlassen wird

Gespeichert wird nicht permanent, also auch nicht über mehrere Online-Sitzungen.

Pro Objekt wird dazu ein objekttegener User-Data-Store verwendet.

Syntax:

<META NAME="save" CONTENT="history">

sowie

XML <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#saveHistory ')" >

HTML <ELEMENT STYLE="behavior:url('#default#saveHistory')" ID=kette>

Scripting object.style.behavior = "url('#default#saveHistory')"

object.addBehavior ("#default#saveHistory")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

.....>

....

</.....>

Prefix laut XML-Namensraum

CustomTag User-definierter Tag

kette String mit dem ID
muss kodiert werden

Die META-Angabe **muss** kodiert werden (im HEAD) !

Beispiel:

```

<HTML>
<HEAD>
    <META NAME="save" CONTENT="history">
    <STYLE>
        .HistoryKlasse { behavior:url('#default#saveHistory');}
    </STYLE>

```



```

<SCRIPT>
    function InterneDateSpeichern()
    {
        // Textwert des Input als neues Attribut erzeugen und speichern
        ID_Input.setAttribute("InterneDateAlsAttribut", ID_Input.value);
    }

    function InterneDateLaden()
    {
        // Textwert des Input laden
        ID_Input.value= ID_Input.getAttribute("InterneDateAlsAttribut");
    }
</SCRIPT>
</HEAD>
<BODY>
    <A HREF="www.test.de">
        Dokument verlassen
    </A>
    <INPUT TYPE=text
        ID="ID_Input"
        CLASS= FavoriteKlasse
        onsave="InterneDateSpeichern()"
        onload="InterneDateLaden()"
    >
</BODY>
</HTML>

```

Eigenschaften:

.XMLDocument Referenz auf XML-Dokument (XML-DOM)

Methoden:

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
DOM nicht geändert

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
DOM wird geändert

.setAttribute() Wert von vorhandenem Attribut setzen
wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
DOM wird nur bei Erzeugung geändert

4.3.2.2.4.3.39.23.8. .style.saveSnapshot Behavior des Internet Explorer

frei definierbare Daten aus dem Dokument sichern , wenn User das Dokument per Menüpunkt "Datei-Speichern (unter) " als **HTML-Datei** auf die Festplatte ablegt, also einen HTML-Schnappschuss des Dokumentes erzeugt.

ab IE 5.x

siehe auch document.namespace Objekt

Es wird folgendes Event vom Objekt **mit** dem saveSnapshot-Behavior ausgelöst:

onsave wenn Dokument gespeichert wird auf Festplatte als HTML-Datei

Gespeichert wird nicht permanent, also auch nicht über mehrere Online-Sitzungen.

Pro Objekt wird dazu ein objekteneigener User-Data-Store verwendet.

Gespeichert werden können folgende Objekte **nicht**: body Objekt
alle Tabellenelemente (Objekte), die unterhalb dem table Objekt liegen
wie Zelle oder Spalte

Gespeichert werden können **nur** folgende Datentypen:

String, Boolean und Integer-Varianten

also **keine** Zeiger, Felder als Ganzheit (elementweise aber speicherbar wenn Typ wie oben) etc.

Syntax:

<META NAME="save" CONTENT="snapshot">

sowie

XML <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#saveSnapshot')">
HTML <ELEMENT STYLE="behavior:url('#default#saveSnapshot') ID=kette>
Scripting object.style.behavior = "url('#default#saveSnapshot')"
object.addBehavior ("#default#saveSnapshot")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>
.....</>

sollte ein eigenständiges Endetag kodiert werden, also



Prefix	laut XML-Namensraum
CustomTag	User-definierter Tag
kette	String mit dem ID
	muss kodiert werden

Die META-Angabe **muss** kodiert werden (im HEAD) !

Beispiel:

```
<HTML>
<HEAD>
  <META NAME="save" CONTENT="snapshot">
  <STYLE>
    .DateiSpeichernUnterKlasse { behavior:url(#default#saveSnapshot); }
  </STYLE>
  <SCRIPT>
    function InterneDateSpeichern()
    {
      // Textwert des Input als neues Attribut erzeugen und speichern
      ID_Input.setAttribute("InterneDateAlsAttribut", ID_Input.value);
    }
  </SCRIPT>
</HEAD>
<BODY>
  <INPUT TYPE=text
    ID="ID_Input"
    CLASS= DateiSpeichernUnterKlasse
    onsave="InterneDateSpeichern()"
  >
</BODY>
</HTML>
```

Eigenschaften:

keine

Methoden:

keine

4.3.2.2.4.3.39.23.9. .style.time2 Behavior des Internet Explorer

verwaltet die Zeitsteuerung (Timeline, Zeitlinie) für Elemente eines HTML-Dokumentes wie z.B.

Sound,
Videos
Nicht-Media-Objekten
in Verbindung mit transitionFilter (siehe Objekt filter)

anhand von vordefinierten Time-Containern als Timer, die mehrere Elemente im Dokument (Elemente-Gruppe) steuern können
(Damit entfällt die Programmierung von Timern von Hand per Javascriptmethoden wie .setTimeout() etc.)

anhand beliebig vieler Timer im Dokument

anhand von vordefinierten Time-Container zur Art der Animation von Elementgruppen z.B. sequentiell oder parallel

anhand von vordefinierten Eigenschaften des time2 Behavior zur Vor – und Rückwärtsanimation eines Elementes
auf der Timeline.

anhand von XML-Komponenten (vordefinierte Tags im Format **t:**vordefinierter_name_des_container)

komplett erst ab IE 6.x unterstützt

teilweise ab IE 5.5 unterstützt

Behavior time2 ersetzt Behavior time, weil das time deprecated ist.
benötigt keine *.htc-Datei (HTC-Datei) da im Browser als Standard-Behavior implementiert
erst mit kompletten Laden des Dokumentes verwendbar (window.onload Event muss ausgelöst sein).

Die Unterstützung von Eigenschaften und Methoden eines Objektes bezüglich der Timeline-Steuerung ist objektspezifisch (siehe jeweilige Beschreibung der Objekte).

siehe auch timeChildren Collection und document.body.timeAll Collection

4.3.2.2.4.3.39.23.9.1. Timer Konzept des Internet Explorer

4.3.2.2.4.3.39.23.9.1.1. Ansatz

Anhand der Timed Interactive Multimedia Extensions (zeitgesteuerte interaktive Multimedia Erweiterungen)
wird in HTML die zeitliche Synchronisation von Media- und Nichtmedia-Objekten möglich (z.B. Sound, Video, DIV-Objekt).

Die Time-Gruppierung von HTML-Elementen im IE erfolgt per XML-Tags (siehe Objekt transitionFilter) oder der dem .timeContainer Attribut des .style.time2 Behavior. Basissprache ist Synchronized Multimedia Integration Language (SMIL) 2.0.

Timer sind verschachtelbar.

Zur Zeitsteuerung innerhalb der Gruppe wird die Zeitlinie (Timeline) des jeweiligen Objektes benutzt, die auch vererbt wird. Die Timeline ist somit die Abfolge von Zeitpunkten der Animation des jeweiligen Objektes.



Beispiel: Die Timeline des Dokumentes startet, sobald das Dokument geladen wurde. Das Anzeigen (Rendern) der Dokument-Elemente erfolgt ebenfalls in der Timeline. Diese Timeline kann mit denen der Objekte im Dokument synchronisiert werden.

Zur Synchronisation der Timelines in der Gruppe und zur Interaktion zwischen den Gruppenobjekten werden Events genutzt. Vererbung und Events ermöglichen die zeitliche Synchronisation der Timelines abhängiger Objekte. Nicht jedes Event wird von einem Kind an die Eltern durchgereicht.

Die Interaktion mit dem User ist anhand der Timeline ebenfalls möglich.

Die Timeline einer Media-Datei, die erst gedownload werden muss, kann erst **nach** dem kompletten Download aktiv werden. Das Ende des Downloads wird vom time2 Behavior nicht erkannt, muss also programmiert werden per Eventereignis (siehe auch Behavior .style.download).

Die Timeline des Dokumentes ist die Timeline des document.body Objektes.

Die Unterstützung von Eigenschaften und Methoden eines Objektes bezüglich der Timeline-Steuerung ist objektspezifisch (siehe jeweilige Beschreibung der Objekte).

Die Timeline kann verschiedene Zustände haben:

unbekannt, inaktiv, aktiv	
cueing	Timeline verarbeitet media file
	Hinweis: Element ist aktiv
	nur für Element das media file verarbeiten kann
seeking	Timeline sucht einen Punkt im media file
	Hinweis: Element ist aktiv
	nur für Element das media file verarbeitet
holding	Timeline ist gehalten
	Hinweis: Element ist nicht aktiv, aber Element wartet auf Ende der Timeline der Eltern

Ein Objekt (Element) kann auf seiner Timeline verschiedene Zustände bezüglich der Eventverarbeitung haben:

aktiv	Element kann Events verarbeiten
inaktiv	ohne holding Timeline: Element kann keine Events verarbeiten
inaktiv	bei holding Timeline: Element kann nur das folgende Event verarbeiten:
	Warten auf das Ende der Timeline der Eltern

Es ist möglich, ein Element auf der Timeline rückwärts zu animieren, aber rückwärts nur bezüglich Timeline.

Ein Video-Element rückwärts abzuspielen, setzt voraus, dass frameweise auf der Timeline animiert wird, also die Timeline die Framefolge bestimmt.

Typisches Beispiel für Rückwärtsanimation ist z.B. eine Gruppe von DIV's innerhalb t:SEQ (sequentieller Time-Container), deren sequentielle Reihenfolge der Anzeige einmal vorwärts oder rückwärts sein kann (vorwärts entspricht Kodierungsfolge der DIV's innerhalb des Time-Containers).

4.3.2.2.4.3.39.23.9.1.2. Alternative zum Timer Konzept des IE

Alternativ zum Time-Konzept kann die selbstprogrammierte Zeitsynchronisation verwendet werden, die **nicht nur** dem IE vorbehalten ist. Dazu wird im Dokument als "Herz" eine zentrale Timer-Funktion implementiert, innerhalb derer die Rückkehrcode der Objekte des Dokumentes ausgewertet werden. Anhand dieser Code ist Synchronisation möglich. Die Objekte können selbst Timer-Funktionen haben, müssen aber ihre Ereignisse dem "Herz" melden, das mit dem Laden des Dokumentes aktiviert werden muss. Als Ereignisse werden globale Variablen verwendet, deren Belegung auf einen definierten Wert durch das "Herz" periodisch abgefragt werden. Das "Herz" muss damit eine rekursive Funktion sein, die sich selbst im Zeitintervall aufruft.

Man beachte: Je mehr Timer implementiert sind, um so ungenauer geht die PC-Uhr, da Windows selbst Timer benutzt und diese mit den Dokument-Timern teilen muss. Die PC-Hardware bietet nur begrenzt Timer an, die von der Software benutzt werden.

4.3.2.2.4.3.39.23.9.1.3. XML-Kodierung in HTML

Nachfolgende Beschreibung hat Einführungscharakter und ist nicht umfassend. Die umfassenden konkreten Kodierungsformen eines Objektes zum .style.time2 Behavior sind der jeweiligen Objektbeschreibung zu entnehmen. Die konkrete Kodierung zum .style.time2 Behavior ist aus dessen Beschreibung zu entnehmen.

Namensraum:

Hinweis: Nachfolgend fett dargestellte Kodierungsteile sind Pflichtkodierung, also nicht zu verändern.

erzeugen und importieren

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
    <?IMPORT          namespace="t"
                      implementation="#default#time2"
    >
oder
    <?IMPORT          namespace = t
                      urn = "urn:schemas-microsoft-com:time"
                      implementation = "#default#time2"
    >
    .....
</HEAD>

```



referenzieren

```

<html_element_tag STYLE="behavior:url(#default#time2)" ...>

oder

<HTML>
<HEAD>
    <STYLE>
        .freier_klassen_name { behavior: url(#default#time2); }
    </STYLE>
</HEAD>
<BODY>
    ....
    <html_element_tag CLASS="freier_klassen_name" ...>
</BODY>
</HTML>

```

erzeugen, importieren und referenzieren

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
</HEAD>
<BODY>
    ....
    <html_element_tag STYLE="behavior:url(#default#time2)" ...>
    ....
</BODY>

oder

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .freier_klassen_name { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    ....
    <html_element_tag CLASS="freier_klassen_name" ...>
    ....
</BODY>

```

XML-Tag im Namensraum des Behavior:

sind vordefinierte Tags vorhanden (haben immer Endetag), die eine Timerart repräsentieren.

Bsp:

```

t:EXCL
t:SEQ
t:PAR
t:AUDIO
t:VIDEO

```

Der Namensraum **t** muss erzeugt sein und wird durch den Buchstaben vor dem Doppelpunkt referenziert.

XML-Tags des Behavior sind z.T. verschachtelbar.

4.3.2.2.4.3.39.23.9.1.4. Timeline für ein Objekt anhand HTML-Attribute erzeugen (BEGIN, END, DUR, TIMEACTION)

Objekt kann HTML- oder XML-Element sein

Attribute stammen aus dem Behavior style.time2 (siehe dort, z.T. auch in der Beschreibung anderer Objekte mit genannt z.B. DIV-Objekt)

Beginn der Timeline:

Attribut BEGIN
Wert in Sekunden als String
Wartezeit nach dem Laden des Elternobjektes

Ende der Timeline:

Attribut END oder Attribut DUR (Dauer)
Wert in Sekunden als String
für END: absoluter Zeitpunkt
für DUR: relativ zu BEGIN

Aktion während der Timeline:

Attribut TIMEACTION mit Zeichenkettenwert

Wert abhängig vom Element

oder auf "none" gesetzt, so trotzdem in den Kinder definierte Aktionen

ausführbar, wenn Eltern-Timeline aktiv ist

nur von einem aktiven Element ausführbar und wenn Eltern- und Element-Timeline aktiv sind

Beispiel für HTML-Elemente:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<P ID="ID_P1"
CLASS="time-line_klasse"
BEGIN="2"
DUR="5"
>
Test
</P>
<P ID="ID_P2"
CLASS="time_line_klasse"
BEGIN="3"
DUR="6"
TIMEACTION="display"
>
Test
</P>
<P ID="ID_P3"
CLASS="time_line_klasse"
BEGIN="4" DUR="7"
TIMEACTION="visibility"
>
Test
</P>
<P ID="ID_P4"
CLASS="time_line_klasse"
BEGIN="1"
DUR="3"
TIMEACTION="hidden"
>
Test
</P>
<P ID="ID_P5"
CLASS="time_line_klasse"
BEGIN="1"
DUR="3"
TIMEACTION="none"
>
Test
</P>
<H1 ID="ID_H1"
CLASS="time_line_klasse"
BEGIN="0"
DUR="11"
TIMEACTION="style"
STYLE="Color:Red;"
>
Test
</H1>
</BODY>
</HTML>
```

4.3.2.2.4.3.39.23.9.1.5. Timeline für ein Objektgruppe (Timecontainer) erzeugen (Attribut TIMECONTAINER)

Objekt kann HTML- oder XML-Element sein (siehe Timeline für Objekt)

Die Objektgruppe wird als Time Container bezeichnet.

Innerhalb des Containers können Objekte z.B. wie folgt synchronisiert werden:

t:EXCL exklusiv Timeline der Elemente nicht synchronisieren

t:SEQ sequentiell Timeline der Element als Gesamtfolge aus 1 sequentiellen

Elemente-Animation



t:PAR parallel

Timeline der Element als Gesamtfolge aus parallelen
Elemente-AnimationenAttribute der Objektgruppe:
wie die eines Objektes

Anstelle des Attributes TIMECONTAINER können vordefinierte XML-Tags verwendet werden

z.B. t:SEQ
t:PAR
t:EXCL

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
  <?IMPORT namespace="t" implementation="#default#time2">
  <HEAD>
  <STYLE>
    .time_line_klasse {behavior: url(#default#time2);}
  </STYLE>
  </HEAD>
  <BODY>
    <t:SEQ ID="ID_Seq">
      <DIV ID="ID_Div1" CLASS="time_line_klasse" DUR="2">Div1</DIV>
      <DIV ID="ID_Div2" CLASS="time_line_klasse" DUR="2">Div2</DIV>
      <DIV ID="ID_Div3" CLASS="time_line_klasse" DUR="2">Div3</DIV>
      <DIV ID="ID_Div4" CLASS="time_line_klasse">Div4</DIV>
    </t:SEQ>
  </BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.1.5.1. Timeline für eine Objektgruppe aus einem unverschachtelten Timer

Das Attribut TIMECONTAINER referenziert den Timer-Typ.

Beispiel 1 Marquee aus Bildern

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
  <HEAD>
  <?IMPORT namespace="t" implementation="#default#time2">
  <STYLE>
    .time_line_klasse { behavior: url(#default#time2);}
  </STYLE>
  </HEAD>
  <BODY>
    <MARQUEE ID="ID_Marquee"
      CLASS="time_line_klasse"
      TIMECONTAINER="seq"
      REPEATCOUNT="indefinite"
    >
      <IMG ID="ID_Img1" CLASS="time" DUR="4" SRC="test1.gif" ALT="Test1">
      <IMG ID="ID_Img2" CLASS="time" DUR="4" SRC="test2.gif" ALT="Test2">
      <IMG ID="ID_Img3" CLASS="time" DUR="4" SRC="test3.gif" ALT="Test3">
    </MARQUEE>
  </BODY>
</HTML>

```

Beispiel 2 Tabelle animieren:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
  <HEAD>
  <?IMPORT namespace="t" implementation="#default#time2">
  <STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
    .TextAufHighLight{ color:#CCCCC; font-weight:bold;}
  </STYLE>
  <SCRIPT>
    function PauseAufheben()
    {
      ID_Table.resumeElement();
      ID_Button1.disabled = false;
      ID_Button2.disabled = true;
    }

    function Pausieren()
    {
      ID_Table.pauseElement();
      ID_Button1.disabled = true;
    }
  </SCRIPT>

```



```

        ID_Button2.disabled = false;
    }
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
    <TABLE ID="ID_Table"
        BORDER="1"
        CLASS="time_line_klasse"
        DUR="8s"
        REPEATCOUNT="indefinite"
        TIMECONTAINER="SEQ"
        TIMEACTION="none"
    >
    <TBODY>
        <TR>
            <TH WIDTH="120">Eins</TH>
            <TH WIDTH="50">Zwei</TH>
            <TH WIDTH="40">Drei</TH>
        </TR>
        <TR
            ID="ID_TR1"
            CLASS="time_line_klasse"
            TIMEACTION="class: TextAufHighLight "
            DUR="2s"
        >
            <TD>EinsA</TD>
            <TD>ZweiA</TD>
            <TD>DreiA</TD>
        </TR>
        <TR
            ID="ID_TR2"
            CLASS="time_line_klasse"
            TIMEACTION="class: TextAufHighLight "
            DUR="2s"
        >
            <TD>EinsB</TD>
            <TD>ZweiB</TD>
            <TD>DreiB</TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<BUTTON
    ID="ID_Button1"
    onclick="Pausieren();"
>
    pausieren
</BUTTON>
<BUTTON
    ID="ID_Button2"
    DISABLED="true"
    onclick="PauseAufheben();"
>
    Pause aufheben
</BUTTON>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.1.5.2. Timeline für eine Objektgruppe aus verschachtelten Timern

Ziel von verschachtelten Timern ist, dass die Kindertimer das Element des Eltern-Timer **ebenfalls** animieren. Die Kindertimer müssen also zum Eltern-Timer passen. Ideal dafür sind z.B. Filter, die als Kinder ein Bild im Eltern-Timer mitanimieren.

Das Attribut TIMECONTAINER referenziert den Timer-Typ.

Das Attribut TIMECONTAINER **muss** in **jedem** Timer, der einen Eltern-Timer hat, kodiert werden, damit der Elterntimer weiss, wie die Kinder das Elternelement mitanimieren. Das ist nötig, um verschachtelte Timer synchronisieren zu können. Der Wert des Attributes im Kind-Timer, also der Typ der Animation per Kind-Timer, muss zur Animation per Eltern-Timer passen. Dafür muss der Programmierer sorgen.

Beispiel 1: Im Beispiel sind die STYLE-Attributangaben zur Position (z.B. left etc.) nur aus Übersichtlichkeit nicht kodiert worden.

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

```



```

<DIV ID="ID_Div" STYLE="height:100px">
<t:SEQ ID="ID_Seq"
  REPEATCOUNT="indefinite"
>
  <t:MEDIA ID="ID_Media1"
    SRC = "test1.jpg"
    STYLE="position:absolute; ....."
    DUR="3"
    TIMECONTAINER="par"
    FILL="transition"
  >
    <t:TRANSITIONFILTER ID="ID_Transfilter1"
      TYPE="fade"
      DUR="2"
    >
    </t:TRANSITIONFILTER>
    <t:TRANSITIONFILTER ID="ID_Transfilter2"
      TYPE="ClockWipe"
      DUR="2"
    >
    </t:TRANSITIONFILTER>
  </t:MEDIA>
  <t:MEDIA ID="ID_Media2"
    SRC = "test2.jpg"
    STYLE="position:absolute; ....."
    DUR="3"
    TIMECONTAINER="par"
    FILL="transition"
  >
    <t:TRANSITIONFILTER ID="ID_Transfilter3"
      TYPE="ClockWipe"
      DUR="2"
    >
    </t:TRANSITIONFILTER>
    <t:TRANSITIONFILTER ID="ID_Transfilter4"
      TYPE="fade"
      DUR="2"
    >
    </t:TRANSITIONFILTER>
  </t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>

```

Beispiel 2: Ein Marquee-Objekt als Kindtimer neben parallelen Filtern (ob Marquee und Filter zueinander und zum Eltern-Timer passen ?)
 Im Beispiel sind die STYLE-Attributangaben zur Position (z.B. left etc.) nur aus Übersichtlichkeit nicht kodiert worden.

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
  <DIV ID="ID_Div" STYLE="height:100px">
    <t:SEQ ID="ID_Seq"
      REPEATCOUNT="indefinite"
    >
      <t:MEDIA ID="ID_Media1"
        SRC = "test1.jpg"
        ALT="Test1"
        STYLE="position:absolute; ...."
        DUR="3"
        TIMECONTAINER="par"
        FILL="transition"
      >
        <t:TRANSITIONFILTER ID="ID_Transfilter1"

```



```

        TYPE="fade"
        DUR="2"
    >
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter2"
    TYPE="ClockWipe"
    DUR="2"
>
</t:TRANSITIONFILTER>

<MARQUEE ID="ID_Marquee"
    CLASS="time_line_klasse"
    TIMECONTAINER="seq"
    REPEATCOUNT="indefinite"
    STYLE="position:absolute; ...."
>
    <IMG ID="ID_Img1"
        SRC="test2.gif"
        CLASS="time"
        DUR="4"
        ALT="Test2"
    >
    <IMG ID="ID_Img2"
        SRC="test3.gif"
        CLASS="time"
        DUR="4"
        ALT="Test3"
    >
    <IMG ID="ID_Img3"
        SRC="test4.gif"
        CLASS="time"
        DUR="4"
        ALT="Test4"
    >
    <IMG ID="ID_Img4"
        SRC="test5.gif"
        CLASS="time"
        DUR="4"
        ALT="Test5"
    >
</MARQUEE>
</t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.1.6. Time Formate zum .style.time2 Behavior des Internet Explorer

für Eigenschaften wie z.B. .begin .dur .end .repeatDur und .syncTolerance

"h:m:s.f" h Stunde als Integer
 m Minute als Integer
 s Sekunde als Integer oder Float
 f Hundertstel Sekunden
 jedes Nachfolge-Element kann entfallen

Beispiele "25:45:10" 25 Stunden, 45 Minuten, 10 Sekunden
 "45:35" 45 Minuten, 35 Sekunden
 "45:00.275" 45 Minuten, 0,275 Sekunde
 "10.5" 10,5 Sekunden

"xh" oder "xm" oder "xs" oder "xms"
 xh Stunde mit x Integer
 xm Minute mit x Integer
 xs Sekunde mit x als Integer oder Float
 xms Millisekunde mit x als Integer

"jjjj-mo-ddThh:mm:ss+hh:mm.TZD"

festkodiert sind - T : + .
 jjjj Jahr als Integer 4 stellig
 mo Monat als Integer 2 stellig, mit Vornull
 dd Tag als Integer 2 stellig, mit Vornull
 hh Stunde als Integer 2 stellig, mit Vornull, 00 bis 23
 mm Minute als Integer 2 stellig, mit Vornull, 00 bis 59
 ss Sekunde als Integer oder Float, ohne Vornull



T Trenner zwischen Datum und Uhrzeit
+hh:mm Offset der Stunde und Minute zur Zeit laut TZD
.TZD optional, Platzhalter für Zeitzone, Standard ist UTC (Weltzeit)
 z.B. UTC oder GMT

nur für Eigenschaften .begin und .end kodierbar

objekt_zeiger.event solange Warten bis Ereignis laut event für das Objekt laut id eintritt und dann das Objekt starten
 event entspricht dem Eventbezeichner on "on"
 Bsp: anstelle onclick nur click kodieren
 onbegin nur begin kodieren
 onend nur end kodieren

nur für Eigenschaften .begin und .end kodierbar

id.event+zeit_wert_als_string zeit_wert_als_string siehe oben
 solange Warten bis Ereignis laut event für das Objekt laut id eintritt **und** dann die Wartezeit abwarten
 und dann das Objekt starten
 Standard ist "+0" in Sekunden und kein Event

nur für Eigenschaften .begin und .end kodierbar

Beispiele für Kodierung von Time in der Eigenschaft .begin:

```
object.begin=" objekt_zeiger.begin+10s"      // warten bis Event "onbegin" zum Objekt laut
                                                 //                                                   objekt_zeiger (z.B. laut ID-Attribut)
                                                 //                                                   eintritt,
                                                 //                                                   dann 10 Sekunden warten
                                                 //                                                   dann Objekt laut object starten
object.begin=" objekt_zeiger.focus+10s"      // warten bis Event "onfocus" zum Objekt laut
                                                 //                                                   objekt_zeiger (z.B. laut ID-Attribut)
                                                 //                                                   eintritt,
                                                 //                                                   dann 10 Sekunden warten
                                                 //                                                   dann Objekt laut object starten

object.begin="2; objekt_zeiger.click+1"      // 2 Sekunden nach dem Laden des Elternobjektes von
                                                 //                                                   object warten
                                                 //                                                   dann auf das Ereignis click zum Objekt laut
                                                 //                                                   objekt_zeiger warten
                                                 //                                                   dann 1 Sekunde warten
                                                 //                                                   dann Objekt laut object starten
```

Hinweis: object laut ID-Attribut des Behavior-Objektes von .style.time2.

4.3.2.2.4.3.39.23.9.1.7. Alternativen zum Behavior .style.time2

Alternativen sind: Objekt bgound
 Windows Media Player ab 7.1 (Hinweis: Versionen vor 7.1 sind z.T. sehr inkompatibel zur Version 7.1)

Vergleich .style.time2 Behavior und Windows Media Player ab 7.1:

Die Objekte und Collectionen zum Player und zum Behavior .style.time2 sind konzeptionell ähnlich.
 Der Player und .style.time2 basieren auf gemeinsamen Prinzipien, nur dass sich die referenzierten Objekte und Collectionen anders nennen bzw. verschieden aufgebaut sind.
 Die Eigenschaften zum .style.time2 Behavior und zum Windows Media Player sind sehr ähnlich, wenn nicht gar identisch.

Während der Windows Media Player alle Windows-kompatiblen Medienarten unter einem gemeinsamen Dach verwalten kann, muss beim .style.time2 Behavior das richtige Behavior-Objekt gewählt werden, um entsprechende spezielle Methoden zum Typ des Mediums ansprechen zu können.

Das Objektmodell des Windows Media Players ist wesentlich ganzheitlicher, dadurch übersichtlicher, und hat einen größeren Umfang als das des Behavior. Z.B. kann der Player CD-Laufwerke als Systemkomponente verwalten.

Die Programmierung von .style.time2 ist durch die Verwendung des Players unnötig, setzt aber voraus, dass der Player auch verwendet werden soll.

Es ist zu vermuten, dass die Verwendung des Behavior ressourcenschonender ist, denn es werden nur Eigenschaften und Methoden instanziiert, die für das Behavior-Objekt nötig.

Per .style.time2 Behavior kann der Windows Media Player eingebunden werden, muss aber nicht. Wird der Player eingebunden, so ist er per Behavior ansprechbar. Die Programmierung des Players setzt **aber** eine permanente Player-Instanz im HTML-Dokument voraus und damit ein permanentes Playerfenster. Außerdem wird der Player im **HTML-Dokument** nicht mit allen seine Objekteigenschaften unterstützt, was vielleicht doch wieder für die Nutzung des Behavior .style.time2 spricht.

Behavior .style.time2 wird per XML-Tag in das HTML-Dokument eingebunden. Der Player wird in das HTML-Dokument per Active-X-Control im OBJECT-Tag eingebunden.



Es ist zu vermuten, dass der Behavior ebenfalls das Active-X-Control benutzt, wenn der Windows Media Player in den Behavior eingebunden ist.

Damit wird klar, warum es kein Plugin für den Netscape zum Windows Media Player 7.1 mehr gibt:

Ein Plugin ist kein Active X-Control. Microsoft überlässt es den Fremdanbietern, ihre Playersoftware in Windows 32 Bit per Active-X-Control zu implementieren, da ab IE 6.x generell keine Plugins mehr unterstützt werden. Vorteil ist die Normung per Active-X-Schnittstelle und somit die Komplettintegration des Players und IE in Windows. Nachteil ist das Ausbieten von Konkurrenzprodukten, wenn Microsoft die notwendigen Schnittstellen nicht komplett offengelegt hat und damit eine Weiterentwicklungen durch die Konkurrenz vor allem für Anwender verhindert, die nicht unbedingt mit der Microsoft-Player-Software arbeiten wollen bzw. andere Features erwarten, also sie Microsoft bisher implementiert hat.

4.3.2.2.4.3.39.23.9.1.8. Medien zur Animation per Behavior .style.time2

Media Datei kann lokal auf dem User-PC oder im Netzwerk (z.B. Internet) liegen
Video und Audio in diversen Formen z.B.:

nicht Windows-spezifisch:	*.AVI *.MID *.MP3 *.MPEG *.WAV
Windows-spezifisch	*.WM *.WMA *.WMV *.ASF

Datei-Suffix	MIME type	Windows-Media-Datei
*.wma	audio/x-ms-wma	nur Audio
*.wmv	video/x-ms-wmv	Audio/Video
*.asf	video/x-ms-asf	Audio/Video

hat Media-Typ z.B. "Audio" oder "Video"

kann Attribute folgende Attribute besitzen

"Album"	nur bei Media Item
"Artist"	nicht für Playlist, die per Methode ID_Player.mediaCollection.getByXXX() oder ID_Player.mediaCollection.getAll() erzeugt wurde
"Author"	
"Bitrate"	Bitrate, nur bei Element aus Media-Bibliothek
"Copyright"	nur bei Playlist-Eintrag nicht Playlist von CD nicht Media-Item
"CreationDate"	Datum der Hinzufügen des Elementes zur Media-Bibliothek nicht Playlist und Media Item
"DigitallySecure"	Datum des Schutzes des Elementes in der Media-Bibliothek Schutz = Digital Rights Management nicht Playlist und Media Item
"Genre"	nur Playlist-Eintrag nicht Playlist von CD
"MediaType"	Media-Typ (audio oder video)
"Name"	Name des Playlist-Eintrages
"PlayCount"	Anzahl der Wiederholungen eines Elementes aus der Media-Bibliothek nicht Playlist und Media Item
"SourceURL"	Url oder Pfad und Dateiname der Media-Datei als Element in der Media Bibliothek nicht Playlist und Media Item
"TOC"	CD Table of Contents Identifier nur für Playlist von CD nicht für Playlist, die per Methode ID_Player.mediaCollection.getByXXX() oder ID_Player.mediaCollection.getAll() erzeugt wurde



Die Attribute haben natürlich je nach Media-Datei einen konkreten Inhalt.

Media Clip dasselbe wie Media-Datei, aber nur Video

Meta-Datei Windows-spezifische Script-Datei z.B. ASX-Datei
Die Meta-Elemente in der Datei sind spezifisch zur Art der Meta-Datei.
Jede Meta-Datei kann mindestens 1 Media-Datei und/oder andere Meta-Datei laden:

Bezüglich nicht-Windows-spezifischer Media-Dateien gilt:
Meta-Dateien können sie alle laden.

Bezüglich der Windows-spezifischen Media-Dateien gilt:
Meta-Dateien können nur bestimmte Media-Dateien referenzieren, da
die Kompressionstechnologien nicht in jeder Meta-Datei verfügbar sind.

Übersicht zu Meta-Dateien und ladbaren Windows-spezifischen Media-Dateien

Meta-Datei-Suffix	Windows-Media-Datei-Suffixe		
	*.asf	*.wma	*.wmv
*.wvx	X	X	X
*.wax	X	X	
*.asx	X		

X = ladbar

Übersicht zu Mime-Typen:

Meta-Datei-Suffix	MIME type	Windows-Media-Datei			
*.wax	audio/x-ms-wax	Media-Datei mit Suffix	*.asf	*.wma	
		Meta-Datei mit Suffix		*.wax	
*.wvx	video/x-ms-wvx	Media-Datei mit Suffix		*.wma	*.wmv
		Meta-Datei mit Suffix		*.wax	*.wvx
*.asx	video/x-ms-asf	Media-Datei mit Suffix	*.asf	*.wma	*.wmv
		Meta-Datei mit Suffix	*.asx	*.wax	*.wvx

Playliste ist Liste von mindestens 2 media Objekten aus der Media-Bibliothek
z.B. in Form einer ASX-Meta-Datei
Playliste kann physisch sein lokal oder im Netzwerk
Datenbank oder Textdatei

4.3.2.2.4.3.39.23.9.2. Beschreibung des Behavior

4.3.2.2.4.3.39.23.9.2.1. Syntax

XML <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#time2')">
HTML <ELEMENT STYLE="behavior:url('#default#time2')" ID=kette>
Scripting object.style.behavior = "url('#default#time2')"
object.addBehavior ("#default#time2")

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

.....>

....

</.....>

Prefix XML-Namensraum des Behavior
muss immer kodiert werden als
t:xxx

festkodiert sind t und der Doppelpunkt

xxx vordefinierter Bezeichner eines Objektes zum .style.time2 Behavior
z.B.

animation Animationselement
audio Audio-Element
excl Timeline-Container **nur** für Kinder des Elementes
img Image-Element
media Generisches Element
par Neuer Timeline-Container für unabhängiges Element
ref Referenz auf ein generisches Element
seq Neuer Timeline-Container für sequentielle
Zeitsteuerung eines Elementes
video Video-Element
siehe Objektbeschreibungen



Objekte sind die Timer vom Behavior .style.time2. Diese Timer sind vordefiniert und ermöglichen somit nur vordefinierte Arten von Animationen eines Elementes. Die Gestaltung der Animation ist dem Programmierer überlassen.

CustomTag User-definierter Tag
kette String mit dem ID des Elementes (Objektes), das animiert werden soll
muss kodiert werden

Hinweis: Die Kodierung von

```
<ELEMENT STYLE="behavior:url('#default#time2')" ID=kette>
```

ist nicht zu empfehlen, weil unübersichtlich.

Für alle zu animierenden Elemente und deren Timer sind immer ID-Attribute zu kodieren, auch um die Performance zu erhöhen. ID-Attribute sind auch dann nötig, wenn sie nicht zur Referenzierung benutzt werden. Empfehlung: Generell im Dokument für jedes Element ein ID kodieren.

Wenn in den Beispielen der Beschreibungen zum Behavior und seiner Objekte die ID-Attribute nicht immer vollständig kodiert wurden, so dient das **nur** dem Zweck der Übersichtlichkeit:

Es werden in den Beispielen oft nur diejenigen ID kodiert, die auch tatsächlich der Referenzierung dienen.

4.3.2.2.4.3.39.23.9.2.2. Kodierung des .style.time2 Behavior in HTML

```
<HEAD>
<STYLE>
.time_line_klasse { behavior: url('#default#time2') }
</STYLE>
</HEAD>
<BODY>
<html_element_tag ID="ID_des_html_elementes" CLASS="time_line_klasse" ...>
</BODY>
```

time_line_klasse ist ein freier Bezeichner, der als Wert des CLASS-Attributes im zu animierenden Element dient und somit dem Element einen Timer zuordnet

oder

```
<BODY>
....
<html_element_tag ID="ID_des_html_elementes" STYLE="behavior:url('#default#time2')" ...>
....
</BODY>
```

4.3.2.2.4.3.39.23.9.2.3. Import des Behavior .style.time2 in HTML

Der Import des Behavior per HTML wird am einfachsten wie folgt kodiert:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url('#default#time2') }
</STYLE>
</HEAD>
<BODY>
<html_element_tag ID="ID_des_html_elementes" CLASS="time_line_klasse" ...>
</BODY>
```

time_line_klasse ist ein freier Bezeichner, der als Wert des CLASS-Attributes im zu animierenden Element dient und somit dem Element einen Timer zuordnet

oder

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
</HEAD>
<BODY>
....
<html_element_tag ID="ID_des_html_elementes" STYLE="behavior:url('#default#time2')" ...>
....
</BODY>
```

4.3.2.2.4.3.39.23.9.2.4. Bezug des Timers auf ein Element (Objekt) in HTML



4.3.2.2.4.3.39.23.9.2.4.1. Bezug auf ein Element, das nicht Media-Daten enthält

Wenn das Element nicht Media-Daten enthält, so muss der Timer auf das Element bezugnehmen, damit er weiss, welches Element er animieren soll. Dazu muss das Element die Timerklasse auch referenzieren.

Beispiel für einen Timer per Objekt animate und einem DIV, der mit dem Timer animiert werden soll

Beispiel 1 für genau 1 Element:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:ANIMATE ID="ID_Animate"
TARGETELEMENT="ID_Div"
ATTRIBUTENAME="left"
TO="400"
DUR="3"
ACCELERATE="1"
REPEATCOUNT="3"
>
</t:ANIMATE>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE="position: absolute; left:10px.;">
</DIV>
</BODY>
</HTML>
```

Beispiel 2 für Objektgruppe aus verschachtelten Elementen:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2);}
</STYLE>
</HEAD>
<BODY>
<MARQUEE ID="ID_Marquee"
CLASS="time_line_klasse"
TIMECONTAINER="seq"
REPEATCOUNT="indefinite"
>
<IMG ID="ID_Img1" CLASS="time_line_klasse" DUR="4" SRC="test1.gif" ALT="Test1">
<IMG ID="ID_Img2" CLASS="time_line_klasse" DUR="4" SRC="test2.gif" ALT="Test2">
<IMG ID="ID_Img3" CLASS="time_line_klasse" DUR="4" SRC="test3.gif" ALT="Test3">
</MARQUEE>
</BODY>
</HTML>
```

4.3.2.2.4.3.39.23.2.4.2. Bezug auf ein Element, das nur Media-Daten enthält

Das Element wird **nur** durch den Timer implementiert.

Die Klassendeklaration innerhalb des HEAD per <STYLE> ... </STYLE> kann daher entfallen.

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.optionale_time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function Anzeige()
{
alert('hasAudio: ' + ID_Media.hasAudio);
}
</SCRIPT>
</HEAD>
<BODY>
<t:MEDIA ID="ID_Media"
```



```

SRC="test.wmv"
BEGIN="0"
FILL="remove"
onmediacomplete="Anzeige ();"
>
</t:MEDIA>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.2.4.3. Bezug auf ein Element ohne Media-Daten, das gemeinsam mit Media-Daten animiert werden soll

Wenn das Element zusätzlich zu einer Media-Datei **und mit** der Media-Datei animiert werden soll, dann müssen der Timer das Element und das Element die Timerklasse referenzieren. Dieses Element ist damit synchron zur Media-Datei-Animation. Im Prinzip stellt das eine Gruppe von Elementen dar, denn der Timer muss zwei Objekte animieren.

```

Beispiel: <HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:VIDEO ID="ID_Video"
SRC="test.avi"
SYNCBEHAVIOR="locked"
>
</t:VIDEO>
<SPAN ID="ID_Span"
CLASS="time_line_klasse"
DUR="0.1"
REPEATCOUNT="indefinite"
onrepeat="ID_Span.innerText= ID_Video.hasDownloadProgress;"
>
</SPAN>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.2.5. HTML-Style-Attribut im zu animierenden Element

Für die Animation eines Elementes **muss** das STYLE-Attribut im Element kodiert werden. Die Eigenschaften im STYLE-Attribut dienen der Animation per Behavior .style.time2 und seinen Timer-Objekte. .style.time2 kann nur anhand von Style-Eigenschaften animieren. Das trifft auch für filter Objekte zu, die ja ebenfalls im Style kodiert werden.

4.3.2.2.4.3.39.23.9.2.6. Animation eines Elementes anhand einer speziellen Eigenschaft im HTML-Attribut STYLE

Um optional anhand einer speziellen Style-Eigenschaft eines Elementes zu animieren, muss dem Timer die Style-Eigenschaft wie folgt mitgeteilt werden:

```

<t:ANIMATE ID="ID_Animate"
TARGETELEMENT="ID_Div"
ATTRIBUTENAME="width"
....
>
</t:ANIMATE>

animierter DIV
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE= ".....; width=....."
>
</DIV>

```

4.3.2.2.4.3.39.23.9.2.7. Implementierung Timer bzw. Neusetzung seiner Attribute bei aktivem / nichtaktivem Element

Zur Implementierung eines Timers bzw. Neusetzung seiner Attribute darf das Element (Timeline des Elementes) **nicht** bereits aktiv sein. Element beenden per object.endElement();

Nach der Implementierung des Timers bzw. Neusetzung seiner Attribute muss das Element und seine Timeline (neu) gestartet werden. Element starten per object.beginElement();

4.3.2.2.4.3.39.23.9.2.8. Synchronisierung von Timelines

4.3.2.2.4.3.39.23.9.2.8.1. Synchronisierung von verschachtelten Timelines (verschachtelte Timer)

Ziel von verschachtelten Timern ist, dass die Kindertimer das Element des Eltern-Timer **ebenfalls** animieren. Die Kindertimer müssen also zum Eltern-Timer passen. Ideal dafür sind z.B. Filter, die als Kinder ein Bild im Eltern-Timer mitanimieren.

Das Attribut TIMECONTAINER referenziert den Timer-Typ.



Das Attribut TIMECONTAINER **muss** in **jedem** Timer, der einen Eltern-Timer hat, kodiert werden, damit der Elterntimer weiss, wie die Kinder das Elternelement mitanimieren. Das ist nötig, um verschachtelte Timer synchronisieren zu können. Der Wert des Attributes im Kind-Timer, also der Typ der Animation per Kind-Timer, muss zur Animation per Eltern-Timer passen. Dafür muss der Programmierer sorgen.

Beispiel: Im Beispiel sind die STYLE-Attributangaben zur Position (z.B. left etc.) nur aus Übersichtlichkeit nicht kodiert worden.

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<DIV ID="ID_Div" STYLE="height:100px">
<t:SEQ ID="ID_Seq"
REPEATCOUNT="indefinite"
>
<t:MEDIA ID="ID_Media1"
SRC = "test1.jpg"
STYLE="position:absolute; ....."
DUR="3"
TIMECONTAINER="par"
FILL="transition"
>
<t:TRANSITIONFILTER ID="ID_Transfilter1"
TYPE="fade"
DUR="2"
>
</t:TRANSITIONFILTER>
<t:TRANSITIONFILTER ID="ID_Transfilter2"
TYPE="ClockWipe"
DUR="2"
>
</t:TRANSITIONFILTER>
</t:MEDIA>
<t:MEDIA ID="ID_Media2"
SRC = "test2.jpg"
STYLE="position:absolute; ....."
DUR="3"
TIMECONTAINER="par"
FILL="transition"
>
<t:TRANSITIONFILTER ID="ID_Transfilter3"
TYPE="ClockWipe"
DUR="2"
>
</t:TRANSITIONFILTER>
<t:TRANSITIONFILTER ID="ID_Transfilter4"
TYPE="fade"
DUR="2"
>
</t:TRANSITIONFILTER>
</t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>
```

4.3.2.2.4.3.39.23.9.2.8.2. Zwangs-Synchronisierung von Timelines

Nachfolgendes Beispiel soll die Synchronisierung zeigen zwischen

Timer und genau 1 Element

Timer und Time-Container, wobei der Container eine Elementgruppe einschliesst.

Beispiel für Wiedergabe einer MIDI-Datei per Windows Media Player

MEDIA und ANIMATEMOTION werden per **SYNCBEHAVIOR** zwangssynchronisiert,
wobei der Master der Synchronisation (**SYNCMASTER**) der Timer MEDIA ist.

Ein Master kann **nur** genau einen anderen Timer / Time-Container synchronisieren



(Synchronisationen der Timelines). Der Timer ANIMATEMOTION muss also ohne Attribut **SYNCBEHAVIOR** sein. Um aber trotzdem alle Timer / Time-Container zu synchronisieren, wird das **BEGIN**-Attribut auf den Wert des gleichnamigen Attributes des Masters gesetzt. Der Timer-Container t:SEQ wird durch setzen der Eigenschaft **BEGIN** auf den Wert laut **BEGIN** vom Timer MEDIA ebenfalls synchronisiert. Damit ist schon deswegen der Start der sequentiellen Animation des DIV1 bis DIV 4 identisch mit dem Start von MEDIA.

Die Animation des DIV5 durch ANIMATEMOTION als Timer vom DIV5 (**TARGETELEMENT**="ID_Div5") beginnt ebenfalls mit Start von MEDIA: Der Wert des **BEGIN**-Attributes von ANIMATEMOTION ist identisch mit dem Wert des gleichnamigen Attributes von MEDIA.

Man hätte sich die Kodierung von SYNCMASTER und SYNCBEHAVIOR durch die Wertsetzung der **BEGIN**-Attribute auf den **BEGIN**-Wert von MEDIA sparen können. Die Zwangssynchronisation von MEDIA und dem dem Time-Container SEQ soll hier beispielhaft sein.

Es ist zu beachten, dass Zwangssynchronisation auch Elemente erwartet, die sich ohne Laufzeitprobleme synchronisieren lassen. Eine Synchronisierung kann zur Laufzeit scheitern, was dann ein Rücksetzen der Elemente nachsichziehen muss (ist zu programmieren).

Normalerweise muss zur exakten Behandlung der MIDI-File erst geprüft werden, ob z.B. der Download des MIDI-File erfolgreich erfolgt ist. Erst dann sollte die Midi-Wiedergabe gestartet werden. Um diesen Aufwand zu umgehen, wird in MEDIA der Player als ActiveX-Control verwendet, welcher zugleich Master ist. Damit startet ein zwangssynchronisierter Time-Container auch korrekt mit der Wiedergabe der MIDI-Datei.

Alle Elemente mit Timer-Animation, also alle DIV's, müssen die Timerklasse referenzieren und sind im entsprechenden Timer bzw. Time-Container referenziert.

PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}" ist das ActiveX-Control für den Windows Media Player 7.1 im Internet Explorer. Ab IE 6.x werden keine Plugins mehr unterstützt.

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
    <t:PAR ID="ID_Par"
        SYNCBEHAVIOR="locked"
    >
        <t:MEDIA ID="ID_Media"
            SRC="test.mid"
            PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}"
            SYNCMASTER="true"
            SYNCBEHAVIOR="locked"
            BEGIN="1"
        >
        </t:MEDIA>
        <t:ANIMATEMOTION ID="ID_Animatemotion"
            BEGIN="ID_Media.begin"
            DUR="9"
            TARGETELEMENT="ID_Div5"
            PATH="M 0 0 L 100 300"
            FILL="freeze"
        >
        </t:ANIMATEMOTION>
        <t:SEQ ID="ID_Seq"
            STYLE="font-size:18pt;color:#ff0000"
            SYNCBEHAVIOR="locked"
            BEGIN="ID_Media.begin"
            FILL="freeze"
        >
            <DIV ID="ID_Div1"
                CLASS="time_line_klasse"
                DUR="1.5"
            >
                Diese MIDI-File
            </DIV>
            <DIV ID="ID_Div2"
                CLASS="time_line_klasse"
                DUR="1.5"
            >
```




```

        wird mit dem
    </DIV>
    <DIV ID="ID_Div3"
        CLASS="time_line_klasse"
        DUR="2"
    >
        Windows Media Player
    </DIV>
    <DIV ID="ID_Div4"
        CLASS="time_line_klasse"
        DUR="2"
    >
        abgespielt.
    </DIV>
</t:SEQ>
</t:PAR>
<DIV ID="ID_Div5"
    CLASS="time_line_klasse"
    STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
        top:70;left:10;font-size:18;border-style:solid
    "
>
    animierter DIV waehrend der Wiedergabe der MIDI-File
</DIV>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.2.8.3. Synchronisation von Eltern-Time-Container mit seinen Elementen

Die Eigenschaft `.endSync` beeinflusst das Ende der Animation von Elementen in einem gemeinsamen Time-Container bei Ende der Timeline des Eltern-Time-Containers. Es beschreibt also das Verhalten des Eltern-Time-Containers bezüglich der Animation seiner Kind-Elemente.

Folgende Varianten der Synchronisierung sind möglich:

Eltern-Time-Container endet, wenn alle Kind-Elemente geendet sind.

Wenn Kinder mit endloser Wiederholung: Alle endlos zu wiederholende Kinder werden so lange wiederholt, bis die anderen Kinder ohne Endlos-Animation enden. Danach endet der Eltern-Timer-Container und alle Endlos-wiederholungen werden zugleich gestoppt.

Eltern-Container endet, wenn genau ein Kind-Element geendet ist (egal welches und wann gestartet).

Wenn Kinder mit endloser Wiederholung: Endlos-Wiederholungen eines Kindes können nicht zum Ende des Eltern-Time-Container führen. Der Container muss mindestens 1 Element ohne Endloswiederholung besitzen.

Eltern-Container endet, wenn das Element laut ID-Attribut endet. Dieses Element darf natürlich keine Endloswiederholung besitzen.

Standardgemäß endet der Eltern-Time-Container, wenn alle Kind-Elemente geendet sind.

Wenn Kinder mit endloser Wiederholung: Endlos-Wiederholungen eines Kindes können nicht zum Ende des Eltern-Time-Container führen.

Eltern-Time-Container wartet nie auf das Ende der Kinder, sondern endet immer mit Ende der Timeline des Eltern-Time-Containers. Wenn Kinder mit endloser Wiederholung: Endlos-Wiederholungen eines Kindes werden mit Ende des Eltern-Time-Container sofort gestoppt.

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL ID="ID_Excl"
        ENDSYNC="ID_Div2"
    >
        <DIV ID="ID_Div1"
            CLASS="time_line_klasse"
            BEGIN="0"
            DUR="2"
        >
            Zeile 1
        </DIV>
        <DIV ID="ID_Div2"
            CLASS="time_line_klasse"
            BEGIN="2"

```



```

        DUR="2"
    >
        Zeile 2
    </DIV>
    <DIV    ID="ID_Div3"
        CLASS="time_line_klasse"
        BEGIN="3"
        DUR="2"
    >
        Zeile 3
    </DIV>
</t:EXCL>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.2.9. Beispiele in HTML

Beispiel 1: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts

Es wurden für jedes Element im Dokument ein ID kodiert, also auch dann, wenn das ID keiner Referenzierung dient.

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function SpanInhaltInit(Kette)
    {
        // Zustand
        ID_Span1.innerText = "";

        // Kumulationsart
        ID_Span2.innerText =Kette;

        // Zaehler auf 1
        ID_Span3.innerText ="1";

        // DIV-Text festlegen
        ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerText += " genau 1x kumulativ um ";
            ID_Div.innerText += ID_Animate.by
            ID_Div.innerText += " mal "
            ID_Div.innerText += ID_Animate.repeatCount;
        }
        else
        {
            // keine Wertkumulation von .width
            ID_Div.innerText += ID_Animate.repeatCount;
            ID_Div.innerText += " mal um ";
            ID_Div.innerText += ID_Animate.by
        }

        ID_Div.innerText += " aus";
    }

    function KumulationAus()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("none");
        ID_Animate.accumulate="none";

        // DANACH Animation starten
        ID_Animate.beginElement();
    }

    function KumulationEin()
    {

```



```

        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("sum");
        ID_Animate.accumulate="sum";

        // DANACH Animation starten
        ID_Animate.beginElement();
    }

    function Anzeige()
    {
        ID_Span1.innerText = "Animation ist beendet ";
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE      ID="ID_Animate"
                   TARGETELEMENT="ID_Div"
                   ATTRIBUTENAME="width"
                   BY="150px"
                   DUR="3"
                   REPEATCOUNT="3"
                   BEGIN="indefinite"
                   FILL="freeze"
                   onend="Anzeige();"
                   onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

    animierter DIV
    <DIV      ID="ID_Div"
            CLASS="time_line_klasse"
            STYLE= "position:absolute;
                    top:125px;
                    left:25px;
                    height:100px;
                    width:125px;
                    border:solid black 2px;
                    "
    >
    </DIV>
    <BR>

    Zustand der Animation:
    <SPAN ID="ID_Span1"></SPAN>
    <BR>

    Kumulationsart:
    <SPAN ID="ID_Span2"></SPAN>
    <BR>

    Durchlaufzaehler:
    <SPAN ID="ID_Span3"></SPAN>
    <BR>

    <BUTTON ID="ID_Button1"  onclick="KumulationAus()">Kumulation aus </BUTTON>
    <BUTTON ID="ID_Button2"  onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

Beispiel 2 Beispiel für Wiedergabe per Windows Media Player **und** Synchronisation von Elementgruppen:

MEDIA und ANIMATEMOTION werden per **SYNCBEHAVIOR** zwangssynchronisiert, wobei der Master der Synchronisation (**SYNCMASTER**) der Timer MEDIA ist.

Ein Master kann **nur** genau einen anderen Timer / Time-Container synchronisieren (Synchronisationen der Timelines). Der Timer ANIMATEMOTION muss also ohne Attribut **SYNCBEHAVIOR** sein. Um aber trotzdem alle Timer / Time-Container zu synchronisieren, wird das BEGIN-Attribut auf den Wert des gleichnamigen Attributes des Masters gesetzt.

Der Timer-Container t:SEQ wird durch setzen der Eigenschaft BEGIN auf den Wert laut BEGIN vom Timer MEDIA ebenfalls synchronisiert. Damit ist schon deswegen der Start der sequentiellen Animation des DIV1 bis DIV 4 identisch mit dem Start von MEDIA.

Die Animation des DIV5 durch ANIMATEMOTION als Timer vom DIV5



(**TARGETELEMENT**="ID_Div5") beginnt ebenfalls mit Start von MEDIA:

Der Wert des BEGIN-Attributes von ANIMATEMOTION ist identisch mit dem Wert des gleichnamigen Attributes von MEDIA.

Man hätte sich die Kodierung von SYNCMASTER und SYNCBEHAVIOR durch die Wertsetzung der BEGIN-Attribute auf den BEGIN-Wert von MEDIA sparen können.

Die Zwangssynchronisation von MEDIA und dem dem Time-Container SEQ soll hier beispielhaft sein.

Es ist zu beachten, dass Zwangssynchronisation auch Elemente erwartet, die sich ohne Laufzeitprobleme synchronisieren lassen. Eine Synchronisierung kann zur Laufzeit scheitern, was dann ein Rücksetzen der Elemente nachsichziehen muss (ist zu programmieren).

Normalerweise muss zur exakten Behandlung der MIDI-File erst geprüft werden, ob z.B. der Download des MIDI-File erfolgreich erfolgt ist. Erst dann sollte die Midi-Wiedergabe gestartet werden. Um diesen Aufwand zu umgehen, wird in MEDIA der Player als ActiveX-Control verwendet, welcher zugleich Master ist. Damit startet ein zwangssynchronisierter Time-Container auch korrekt mit der Wiedergabe der MIDI-Datei.

Alle Elemente mit Timer-Animation, also alle DIV's, müssen die Timerklasse referenzieren und sind im entsprechenden Timer bzw. Time-Container referenziert.

PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}" ist das ActiveX-Control für den Windows Media Player 7.1 im Internet Explorer. Ab IE 6.x werden keine Plugins mehr unterstützt.

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
  <t:PAR ID="ID_Par"
    SYNCBEHAVIOR="locked"
  >
    <t:MEDIA ID="ID_Media"
      SRC="test.mid"
      PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}"
      SYNCMASTER="true"
      SYNCBEHAVIOR="locked"
      BEGIN="1"
    >
    </t:MEDIA>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
      BEGIN="ID_Media.begin"
      DUR="9"
      TARGETELEMENT="ID_Div5"
      PATH="M 0 0 L 100 300"
      FILL="freeze"
    >
    </t:ANIMATEMOTION>
    <t:SEQ ID="ID_Seq"
      STYLE="font-size:18pt;color:#ff0000"
      SYNCBEHAVIOR="locked"
      BEGIN="ID_Media.begin"
      FILL="freeze"
    >
      <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        Diese MIDI-File
      </DIV>
      <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        wird mit dem
      </DIV>
      <DIV ID="ID_Div3"
        CLASS="time_line_klasse"
        DUR="2"
      >
        Windows Media Player
```



```

        </DIV>
        <DIV ID="ID_Div4"
            CLASS="time_line_klasse"
            DUR="2"
        >
            abgespielt.
        </DIV>
    </t:SEQ>
</t:PAR>
<DIV ID="ID_Div5"
    CLASS="time_line_klasse"
    STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
        top:70;left:10;font-size:18;border-style:solid
    "
>
    animierter DIV waehrend der Wiedergabe der MIDI-File
</DIV>
</BODY>
</HTML>

```

4.3.2.2.4.3.39.23.9.2.9.

Eigenschaften

.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.autoReverse	automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
.decelerate	Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasMedia	Objekt ist HTML-Media-Objekt
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe .syncTolerance und .syncmaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt bgsound

4.3.2.2.4.3.39.23.9.2.10.

Methoden



<code>.activeTimeToParentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.activeTimeToSegmentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.beginElement()</code>	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.. per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.beginElementAt()</code>	Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.documentTimeToParentTime()</code>	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
<code>.endElement()</code>	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.. per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.endElementAt()</code>	aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.parentTimeToActiveTime()</code>	Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.parentTimeToDocumentTime()</code>	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
<code>.pauseElement()</code>	aktives Element auf Timeline pausieren lassen ersetzt Methode <code>pause()</code> , die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.resetElement()</code>	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
<code>.resumeElement()</code>	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode <code>resume()</code> , da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.seekActiveTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft <code>.canSeek</code>
<code>.seekSegmentTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekTo()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekToFrame()</code>	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.segmentTimeToActiveTime()</code>	Wert der Segment-Timeline des Elements in den korrespondierenden Wert



	der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren
4.3.2.2.4.3.39.23.9.2.11.	Events
onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode .endElement() und Eigenschaft .end
onmediacomplete	erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline siehe onmediaerror
onmediaerror	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf ! siehe onmediacomplete
onpause	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
onrepeat	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement) .repeatCount bzw. .repeat muss > 1 sein: onrepeat wird also .repeatCount - 1 mal erzeugt Kind eines Elementes siehe onend, onbegin, .repeatCount und .repeat
onreset	erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(), also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
onresume	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
onreverse	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist) .autoReverse muss auf "true" stehen
onseek	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame()

4.3.2.2.4.3.39.23.9.2.12. *Objekte und Collectionen zur Verwaltung von style.time2*

4.3.2.2.4.3.39.23.9.2.12.1. *currTimeState Objekt des Internet Explorer*

verwaltet die **aktuelle** HTML- und Zeitsteuerung (Timeline) der Animation eines Elementes und seiner Kinder
Elemente sind z.B. Sound, Videos aber auch Nicht-Media-Objekte

komplett erst ab IE 6.x unterstützt

nur in Verbindung mit dem Objekten time2 (siehe dort) z.B. transitionFilter

Die Kodierung der Eigenschaften muss erfolgen per id_des_objektes.currTimeState.eigenschaft

Hinweis: Objekt time ist deprecated und durch time2 ersetzt worden.



Beispiel 1:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:PAR SPEED="2" >
    <t:KIND ID="ID_Kind"
        BEGIN="1"
        SRC="http://www.test.de/media/test.wmv"
        SPEED="1.5"
    >
    </t:KIND
</t:PAR>
<BUTTON ID="ID_Button1"
    CLASS="time_line_klasse"
    REPEATCOUNT="indefinite"
    onclick="alert( 'Run time speed des Kindes: ' + ID_Kind.currTimeState.speed );"
>
    Run time speed des Kindes
</BUTTON>
<BUTTON ID="ID_Button2"
    CALSS="time_line_klasse"
    REPEATCOUNT="indefinite"
    onclick="alert( 'SPEED-Attribut des Kindes: ' + ID_Kind.speed );"
>
    SPEED-Attribut des Kindes
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<SCRIPT>
function Aendern()
{
    ID_Video.updateMode = ID_Select1.options.value;
    ID_Video.speed      = ID_Select1.options.value;
}
</SCRIPT>
</HEAD>
<BODY>
<t:VIDEO ID="ID_Video"
    SRC="test.avi"
    UPDATEMODE="reset"
    STYLE="width:175px; height:150px;"
>
</t:VIDEO>
<BR>
updateMode waehlen:
<SELECT NAME="ID_Select1">
    <OPTION VALUE="auto">Auto</OPTION>
    <OPTION VALUE="reset" SELECTED>Reset</OPTION>
</SELECT>
<BR>
Geschwindigkeit waehlen:
<SELECT NAME="ID_Select2">
    <OPTION VALUE="0.25">25%</OPTION>
    <OPTION VALUE="0.50">50%</OPTION>
    <OPTION VALUE="0.75">75%</OPTION>
    <OPTION VALUE="1" SELECTED>100% </OPTION>
    <OPTION VALUE="2" SELECTED>200% </OPTION>
</SELECT>
<BR>

```




```
<BUTTON ID="ID_Button1" onClick="Aendern();">
    Geschwindigkeit aendern
</BUTTON>
<BUTTON ID="ID_Button1" onClick="document.body.beginElement()">
    Restart
</BUTTON>
</CENTER>
</BODY>
</HTML>
```

Beispiel 3:

```
<HTML xmlns:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function Stummschaltung()
{
    if (ID_Video.currTimeState.isActive)
    { ID_Span.innerText=' stumm = ' + ID_Video.currTimeState.isMuted;}
    else
    { ID_Span.innerText=' stumm =';}
}
</SCRIPT>
</HEAD>
<BODY>
<t:media ID="ID_Video"
SRC="/test /media/test.wmv"
BEGIN="indefinite;"
FILL="remove"
>
</ t:media>
<BR><BR>
<SPAN ID="ID_Span">stumm =</SPAN>
<BR><BR>
<BUTTON onclick="ID_Video.beginElement();Stummschaltung();">
Start
</BUTTON>
<BUTTON onclick="ID_Video.endElement();Stummschaltung();">
Stop
</BUTTON>
<BR><BR>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio"
onpropertychange=
"ID_Video.mute='true'; Stummschaltung();"
>
Mute
<INPUT TYPE="radio" CHECKED
onpropertychange= "ID_Video.mute='false'; ID_Video.volume=100; Stummschaltung();"
>
100% Volume
</BODY>
</HTML>
```

Eigenschaften:

<u>.activeDur</u>	totale Dauer der Timeline in Sekunden inklusive aller Wiederholungen und Autoreverse
.activeTime	aktueller Zeitpunkt in Sekunden in der Timeline ab Start inklusive aller Repeats, autoReverse
.isActive	Elemente-Status der Aktivität eines Elementes in der Timeline Hinweis: Status der Timeline per Eigenschaft .state oder .stateString abfragen
.isMuted	Elemente-Status der Audio-Stummschaltung in der Timeline
.isOn	Elemente-Status aktiv oder halten in der Timeline Staats aktiv: Objekt kann auf Events reagieren Status halten: Attribut Fill muss auf "freeze" oder "hold" gesetzt sein ermittelbar per eigenschaft .state Objekt kann nicht auf Events reagieren zu Attribut FILL: wenn "hold" oder freeze" dann wartet Element auf die Synchronisation mit der Timeline des Elternobjektes wenn "hold" Element kann keine Events verarbeiten



	wenn "freeze" Element ist aktiv ist inaktiv
.isPaused	Elemente-Status pausieren in der Timeline
.parentTimeBegin	Startzeit als Offset von der Startzeit der Eltern-Timeline
.parentTimeEnd	Endezeit als Offset von der Startzeit der Eltern-Timeline ist zugleich Summe der Werte laut Eigenschaften parentTimeBegin und activeDur
.progress	Hinweis zu activeDur: inklusive aller Wiederholungen und Autoreverse Fortschrittangabe entlang der Timeline inklusive Repeats etc.
.repeatCount	aktuelle Nummer der Wiederholung bei Loop
.segmentDur	Dauer der Timeline der Wiederholungen laut autoReverse
.segmentTime	aktueller Zeitpunkt in der Timeline der Wiederholungen laut autoReverse
.simpleDur	Dauer einer Wiederholung Wiederholung erzeugt per Eigenschaft .autoReverse auf true setzen
.simpleTime	aktueller Punkt auf Timeline
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed) Beispiel: Eltern mit 50 % Geschwindigkeit Kind mit 50 % Geschwindigkeit der Eltern also aktuelle Wiedergabegeschwindigkeit = 25 %
.state	Status der Timeline – Variante 1
.stateString	Status der Timeline – Variante 2
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt currTimeState und Behavior .style.time2 siehe Objekt bgsound

Methoden:

keine

4.3.2.2.4.3.39.23.9.2.12.2. timeChildren Collection des Internet Explorer

Feld aller getimter Kinder eines Elementes

Syntax:

```
[ var FeldZeiger = ] object.timeChildren;

[ var FeldElementZeiger = ] object.timeChildren[Index];
```

Index: Integer und ab 0
muss in [] kodiert sein

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

4.3.2.2.4.3.39.23.9.3. Behavior-Objekte und Behavior-Collections

Nachfolgende Objekte sind die Timer vom Behavior .style.time2. Diese Timer sind vordefiniert und ermöglichen somit nur vordefinierte Arten von Animationen eines Elementes. Die Gestaltung der Animation ist dem Programmierer überlassen.

Als timerorientiertes Objekt ist die Playliste anhand der Objektes playItem und der Behavior-Collection .style.time2.playList zu nenenn.

Zur Implementierung eines Timers bzw. Neusetzung seiner Attribute darf das Element (Timeline des Elementes) nicht bereits aktiv sein.
Element beenden per object.endElement();

Nach der Implementierung des Timers bzw. Neusetzung seiner Attribute muss das Element und seine Timeline (neu) gestartet werden.
Element starten per object.beginElement();

4.3.2.2.4.3.39.23.9.3.1. .style.time2.animate Behavior-Objekt des Internet Explorer

Timerobjekt, das zur Animation eines Elementes anhand einer speziellen Style-Eigenschaft des Elementes dient. Die Style-Eigenschaft muss im STYLE-Attribut des Elementes **und** im ATTRIBUTENAME-Attribut des Timers kodiert sein.

Syntax:

XML t:ANIMATE
Script animate

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
```



```

function SpanInhaltInit(Kette)
{
    // Zustand
    ID_Span1.innerText = "";

    // Kumulationsart
    ID_Span2.innerText = Kette;

    // Zaehler auf 1
    ID_Span3.innerText = "1";

    // DIV-Text festlegen
    ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
    if (Kette == "sum")
    {
        // Wertkumulation von .width
        ID_Div.innerText += " genau 1x kumulativ um ";
        ID_Div.innerText += ID_Animate.by
        ID_Div.innerText += " mal "
        ID_Div.innerText += ID_Animate.repeatCount;
    }
    else
    {
        // keine Wertkumulation von .width
        ID_Div.innerText += ID_Animate.repeatCount;
        ID_Div.innerText += " mal um ";
        ID_Div.innerText += ID_Animate.by
    }

    ID_Div.innerText += " aus";
}

function KumulationAus()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("none");
    ID_Animate.accumulate="none";

    // DANACH Animation starten
    ID_Animate.beginElement();
}

function KumulationEin()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("sum");
    ID_Animate.accumulate="sum";

    // DANACH Animation starten
    ID_Animate.beginElement();
}

function Anzeige()
{
    ID_Span1.innerText = "Animation ist beendet ";
}
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"

```



```

>
<t:ANIMATE>

animierter DIV
<DIV ID="ID_Div"
  CLASS="time_line_klasse"
  STYLE="position:absolute;
        top:125px;
        left:25px;
        height:100px;
        width:125px;
        border:solid black 2px;
        "
>
</DIV>
<BR>

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3"></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

Eigenschaften:

.accelerate Beschleunigung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur
 Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
 siehe .decelerate

.accumulate Kumulative Animation eines Elementes auf der Timeline
 Element darf nicht aktiv sein, wenn .accumulate definiert werden soll:
 Element erst danach starten.
 Es kann jede numerische Style-Eigenschaft kumulativ verändert werden und damit
 die kumulative Animation des Elementes in der Style-Eigenschaft erzeugen.
 Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des
 Wertes sein (mathematische Berechnungen erst nach Entfernung der Einheit
 per Stringoperationen möglich)
 Name der Eigenschaft laut .attributName
 Änderung des Wertes pro Durchlauf bzw. kumulativ um den Maximalwert laut .by
 Anzahl der Durchläufe bzw. der Wert-Kumulationen laut .repeatCount
 Kumulationsschrittweite laut .calcMode
 Ereignis des Startes eines Durchlaufes bzw. Kumulation um Maximalwert ist onrepeat.
 Wert der Eigenschaft .repeatCount muss > 1 sein
 Kumulation nicht aktiv:
 Nach jedem Durchlauf laut .repeatCount erfolgt Rücksetzen der Style-Eigenschaft des
 Elementes auf den Zustand vor dem Durchlaufbeginn.
 Es wird also die Style-Eigenschaft nicht wertmäßig kumuliert.
 Das Element wird also laut .repeatCount mehrmals animiert.
 Pro Durchlauf wird für die Eigenschaft laut .attributName der Wert
 ab Startwert
 in Schrittweite laut .calcMode
 um den maximalen Wert laut .by
 erhöht.

Kumulation aktiv:
 Die Style-Eigenschaft des Elementes wird pro Durchlauf wertmäßig laut
 .repeatCount um den Wert laut .by kumuliert.
 Kumulationsschrittweite laut .calcMode
 Das Element wird also genau 1 mal animiert.
 Mit Beginn des Startes des Elementes wird für die Eigenschaft laut .attributName
 der Wert ab Startwert
 in Schrittweite laut .calcMode
 um den maximalen Wert aus dem Produkt aus
 .by mal Anzahl der Wiederholungen



z.B. laut .repeatCount

	erhöht. Es wird also über alle Wiederholungen der Wert kumuliert.
.additive	siehe .additive .calcMode numerische Werte von Eigenschaften eines Elementes ersetzen oder zu den numerischen Werten der gleichnamigen Eigenschaften anderer Elemente addieren während der Elemente-Animation auf der Timeline Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein gültige Eigenschaften sind z.B. .values oder .by sinnvoll für Kombinationen von Elementen (auch bei Wiederholung eines Elementes) Element darf nicht aktiv sein, wenn .additive definiert werden soll: Element erst danach starten. Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! siehe .accumulate .by .attributeName
.attributeName	Bezeichner einer Style-Eigenschaft, also Name eines Attributes, für die Animation anhand dieses Attributes auf der Timeline
.autoReverse	siehe .accumulate .by .additive .calcMode automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
.by	Wert um den die Werterhöhung bei Elemente-Animation(en) auf der Timeline per .additive oder .accumulate erfolgen soll Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein wird für Animation in Einheiten zerlegt laut Schrittweite laut .calcMode für die Objekte animate, animateMotion und animateColor gilt: .by wird von .path, .to und .values überschrieben Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! siehe .attributeName .accumulate .additive .calcMode
.calcMode	Schrittweite der Werterhöhung bzw. Art der Animation in der Ebene (1D oder 2D) bei Elemente-Animation(en) auf der Timeline per .additive oder .accumulate
.decelerate	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.from	Startwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per .additive oder .accumulate für die Objekte animate, animateMotion und animateColor gilt: .from wird von .path und .values überschrieben Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
.hasMedia	Objekt ist HTML-Media-Objekt
.keySplines	Wert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per .additive oder .accumulate benötigt .calcMode auf "spline" .keyTimes .values oder .path
.keyTimes	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! Zeitwert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per .additive oder .accumulate benötigt .calcMode auf "spline" .keySplines .values oder .path
.repeatCount	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! aktuelle Nummer der Wiederholung bei Loop
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)



.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.targetElement	ID des zu animierenden Elementes auf der Timeline muss für Kindelement immer kodiert werden, wenn nicht das Elternelement animiert werden soll muss nicht kodiert werden, wenn kein Elternelement vorliegt Achtung: Objekt body ist das oberste Elternelement im Dokument und somit haben Elemente innerhalb BODY immer Eltern Empfehlung: immer kodieren ist der Bezug des Time-Container auf das zu animierende Element
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.to	Endwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per .additive oder .accumulate für die Objekte animate, animateMotion und animateColor gilt: .to wird von .path und .values überschrieben .to überschreibt .by Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! 2D-Animation bei der Elemente-Animation(en) auf der Timeline per .additive oder .accumulate anhand absoluter Koordinaten im Grafiksystem siehe .path für Vektorgraphik-Animation anhand relativer und absoluter Koordinaten Werteliste korrespondiert zu der Werteliste der Eigenschaft .keyTimes benötigt .calcMode auf "spline" .keySplines und .keyTimes für die Objekte animate, animateMotion und animateColor gilt: .values wird von .path überschrieben .values überschreibt .by .from .to Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
.values	

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.beginElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben



	falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode .endElement() und Eigenschaft .end siehe onbegin und onrepeat
onpause	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
onrepeat	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement) .repeatCount bzw. .repeat muss > 1 sein: onrepeat wird also .repeatCount - 1 mal erzeugt Kind eines Elementes siehe onend, onbegin, .repeatCount und .repeat
onreset	erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(), also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
onresume	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
onreverse	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist) .autoReverse muss auf "true" stehen



4.3.2.2.4.3.39.23.9.3.2. *.style.time2.animateColor Behavior-Objekt des Internet Explorer*

Timerobjekt, das zur Farbanimation eines Elementes dient. Die zu animierende Farbart ist im STYLE-Attribut des Elementes **und** im ATTRIBUTENAME-Attribut des Timers zu kodieren. Es handelt sich also um eine Animation mit einer speziellen Style-Eigenschaft. zur Farbe.

Syntax:

XML t:ANIMATECOLOR
Script animateColor

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>

<t:PAR ID="ID_Par"
BEGIN="0"
DUR="10"
FILL="hold"

>

<t:ANIMATECOLOR ID="ID_Animatecolor1"
TARGETELEMENT="ID_Div1"
ATTRIBUTENAME="background-color"
VALUES="#0000FF;cyan"
BEGIN="0"
DUR="5"
FILL="hold"

>
</t:ANIMATECOLOR>
<t:ANIMATECOLOR ID="ID_Animatecolor2"
TARGETELEMENT="ID_Div2"
ATTRIBUTENAME="background-color"
VALUES="cyan ;#0000FF"
BEGIN="0"
DUR="5"
FILL="hold"

>

</t:PAR>
<DIV ID="ID_Div1"
CLASS="time_line_Klasse"
STYLE=" position: absolute; left: 68px; width: 279px; top: 260px; height: 217px;
border: 1px solid black; background-color: green;
"

>
animierter Div
</DIV>
<DIV ID="ID_Div2"
CLASS="time_line_classe"
STYLE=" position: absolute; left: 112px; width: 188px; top: 318px; height: 98px;
padding-left: 3; background-color: gray;
"

>
animierter Div
</DIV>
</BODY>
</HTML>
```

Eigenschaften:

.accelerate Beschleunigung des Elementes auf der Timeline
hat keinen Einfluss auf die Dauer der Timeline
auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur
Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten
wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
siehe .decelerate

.accumulate Kumulative Animation eines Elementes auf der Timeline
Element darf nicht aktiv sein, wenn .accumulate definiert werden soll:
Element erst danach starten.
Es kann jede numerische Style-Eigenschaft kumulativ verändert werden und damit
die kumulative Animation des Elementes in der Style-Eigenschaft erzeugen.
Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des
Wertes sein (mathematische Berechnungen erst nach Entfernung der Einheit
per Stringoperationen möglich)
Name der Eigenschaft laut .attributName



Änderung des Wertes pro Durchlauf bzw. kumulativ um den Maximalwert laut `.by`
 Anzahl der Durchläufe bzw. der Wert-Kumulationen laut `.repeatCount`
 Kumulationsschrittweite laut `.calcMode`
 Ereignis des Startes eines Durchlaufes bzw. Kumulation um Maximalwert ist `onrepeat`.
 Wert der Eigenschaft `.repeatCount` muss > 1 sein
 Kumulation nicht aktiv:

Nach jedem Durchlauf laut `.repeatCount` erfolgt Rücksetzen der Style-Eigenschaft des Elementes auf den Zustand vor dem Durchlaufbeginn.

Es wird also die Style-Eigenschaft nicht wertmäßig kumuliert.

Das Element wird also laut `.repeatCount` mehrmals animiert.

Pro Durchlauf wird für die Eigenschaft laut `.attributeName` der Wert

ab Startwert

in Schrittweite laut `.calcMode`

um den maximalen Wert laut `.by`

erhöht.

Kumulation aktiv:

Die Style-Eigenschaft des Elementes wird pro Durchlauf wertmäßig laut

`.repeatCount` um den Wert laut `.by` kumuliert.

Kumulationsschrittweite laut `.calcMode`

Das Element wird also genau 1 mal animiert.

Mit Beginn des Startes des Elementes wird für die Eigenschaft laut `.attributeName`

der Wert ab Startwert

in Schrittweite laut `.calcMode`

um den maximalen Wert aus dem Produkt aus

`.by` mal Anzahl der Wiederholungen

z.B. laut `.repeatCount`

erhöht.

Es wird also über **alle** Wiederholungen der Wert kumuliert.

siehe `.additive .calcMode`

`.additive`

numerische Werte von Eigenschaften eines Elementes ersetzen oder zu den numerischen Werten der

gleichnamigen Eigenschaften anderer Elemente addieren

während der Elemente-Animation auf der Timeline

Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein

gültige Eigenschaften sind z.B. `.values` oder `.by`

sinnvoll für Kombinationen von Elementen (auch bei Wiederholung eines Elementes)

Element darf nicht aktiv sein, wenn `.additive` definiert werden soll:

Element erst danach starten.

Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !

siehe `.accumulate .by .attributeName`

`.attributeName`

Bezeichner einer Style-Eigenschaft, also Name eines Attributes, für die Animation anhand

dieses Attributes auf der Timeline

siehe `.accumulate .by .additive .calcMode`

`.autoReverse`

automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation

des Elementes auf der Timeline

`.begin`

Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft `.timeAction`

`.by`

Wert um den die Werterhöhung bei Elemente-Animation(en) auf der Timeline

per `.additive` oder `.accumulate`

erfolgen soll

Wert der Eigenschaft:

numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein

wird für Animation in Einheiten zerlegt laut Schrittweite laut `.calcMode`

für die Objekte `animate`, `animateMotion` und `animateColor` gilt:

`.by` wird von `.path`, `.to` und `.values` überschrieben

Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !

siehe `.attributeName .accumulate .additive .calcMode`

`.calcMode`

Schrittweite der Werterhöhung bzw. Art der Animation in der Ebene (1D oder 2D)

bei Elemente-Animation(en) auf der Timeline

per `.additive` oder `.accumulate`

Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !

`.decelerate`

Verlangsamung des Elementes auf der Timeline

hat keinen Einfluss auf die Dauer der Timeline

auch wirksam bei Wiederholungen per Attribute `.repeatCount` oder `.repeatDur`

Summe der Werte der Attribute `.accelerate` und `.decelerate` darf nicht 1 überschreiten

wenn ja, so werden beide Attribute ignoriert, also nicht verwendet

siehe `.accelerate`

`.dur`

Dauer Objektaktivitäten laut Eigenschaft `.timeAction`

alternativ: Eigenschaft `.end`

`.end`

Objektaktivitäten laut Eigenschaft `.timeAction` beenden

ab IE 6.x

alternativ: Eigenschaft `.dur`

`.fill`

Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber

bevor die Timeline des Elternelementes endet

ist Ersatz für die Eigenschaft `endHold`, die deprecated ist und nicht mehr verwendet werden darf !!



<code>.from</code>	Startwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> für die Objekte <code>animate</code> , <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.from</code> wird von <code>.path</code> und <code>.values</code> überschrieben Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert !
<code>.hasMedia</code>	Objekt ist HTML-Media-Objekt
<code>.keySplines</code>	Wert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> benötigt <code>.calcMode</code> auf "spline" <code>.keyTimes</code> <code>.values</code> oder <code>.path</code> Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert !
<code>.keyTimes</code>	Zeitwert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> benötigt <code>.calcMode</code> auf "spline" <code>.keySplines</code> <code>.values</code> oder <code>.path</code> Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert !
<code>.repeatCount</code>	aktuelle Nummer der Wiederholung bei Loop
<code>.repeatDur</code>	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft <code>.dur</code> oder <code>.repeat</code> darf nicht kodiert werden mit der Eigenschaft <code>.repeatCount</code>
<code>.restart</code>	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft <code>eventRestart</code> , da diese deprecated ist und nicht mehr verwendet werden darf !!
<code>.speed</code>	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
<code>.systemBitrate</code>	wird hier nicht erklärt
<code>.systemCaptions</code>	wird hier nicht erklärt
<code>.systemLanguage</code>	Sprache festlegen für das Objekt
<code>.systemOverdubOrSubtitle</code>	wird hier nicht erklärt
<code>.targetElement</code>	ID des zu animierenden Elementes auf der Timeline muss für Kindelement immer kodiert werden, wenn nicht das Elternelement animiert werden soll muss nicht kodiert werden, wenn kein Elternelement vorliegt Achtung: Objekt <code>body</code> ist das oberste Elternelement im Dokument und somit haben Elemente innerhalb BODY immer Eltern Empfehlung: immer kodieren ist der Bezug des Time-Container auf das zu animierende Element
<code>.timeContainer</code>	Typ der Timeline des Objektes
<code>.timeParent</code>	Zeiger auf das Eltern-Timeline
<code>.to</code>	Endwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> für die Objekte <code>animate</code> , <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.to</code> wird von <code>.path</code> und <code>.values</code> überschrieben <code>.to</code> überschreibt <code>.by</code> Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert !
<code>.values</code>	2D-Animation bei der Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> anhand absoluter Koordinaten im Grafiksystem siehe <code>.path</code> für Vektorgraphik-Animation anhand relativer und absoluter Koordinaten Werteliste korrespondiert zu der Werteliste der Eigenschaft <code>.keyTimes</code> benötigt <code>.calcMode</code> auf "spline" <code>.keySplines</code> und <code>.keyTimes</code> für die Objekte <code>animate</code> , <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.values</code> wird von <code>.path</code> überschrieben <code>.values</code> überschreibt <code>.by</code> <code>.from</code> <code>.to</code> Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert !
<code>.values</code>	Farbwerte für die Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> nur für Objekt <code>animatecolor</code> (t:ANIMATECOLOR) Achtung: Nur im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! Es existieren als 2 kodierbare Varianten von <code>.values</code> .

Methoden:

<code>.activeTimeToParentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
<code>.activeTimeToSegmentTime()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
<code>.beginElement()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
<code>.beginElementAt()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,



	also ohne Wartezeit
	wenn zusätzlich ein weiterer Start (oder mehrere Starts)
	mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
	mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
	also Neustart nach einem erfolgten Start
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen
	identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.endElementAt()	aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen
	sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen
	ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben
	falls Element nicht pausiert, passiert nichts
	ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	alle Media-Typen für Element zulässig
	siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
	ohne Wiederholung der Animation
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
	einschliesslich möglicher Wiederholungen der Animation
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird
	nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element
	erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element
	siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive



aller Wiederholungen laut `.repeatCount`
 bzw. wenn das aktive Element gestoppt wird
 erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat
 und somit des Kindelement ebenfalls endet
 nicht erzeugt wenn `.fill` mit Wert "freeze" oder "hold" für das Element kodiert wurde,
 es sei denn, das Elternelement hat das Ende seiner Timeline erreicht
 siehe Methode `.endElement()` und Eigenschaft `.end`
 siehe `onbegin` und `onrepeat`
onpause erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren
 auch bei body Objekt
onrepeat erzeugt mit Start **jeder** Wiederholung der Animation des aktiven Elementes
 nicht erzeugt beim Start des aktiven Elementes (siehe `onbegin`), also des
 ersten Durchlaufes, der keine Wiederholung ist
 nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen
 Handler für `onrepeat` kodiert hat (kein Heraufreichen des
 Ereignisses `onrepeat` zum Elternelement)
`.repeatCount` bzw. `.repeat` muss > 1 sein:
`onrepeat` wird also `.repeatCount - 1` mal erzeugt
 Kind eines Elementes
onreset siehe `onend`, `onbegin`, `.repeatCount` und `.repeat`
 erzeugt wenn Element zurückgesetzt wurde per Methode `.resetElement()`,
 also die Timeline des Elementes den aktuellen Wert laut `.begin` wieder erreicht
 und damit zurückgesetzt wurde
 also nicht beim Initialisieren des Elementes und seiner Timeline
 durch Instanziierung des Elementes
onresume erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird
 auch für body Objekt
onreverse erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird
 (auch wenn es sich um eine Wiederholung handelt, also `.repeatCount > 0` ist)
`.autoReverse` muss auf "true" stehen

4.3.2.2.4.3.39.23.9.3.3. *.style.time2.animateMotion Behavior-Objekt des Internet Explorer*

Timerobjekt, das zur 2D-Animation (in der Ebene) des Elementes dient.

Es können nur Style-Eigenschaften des Elementes dazu verwendet werden, die die räumliche Lage des Elementes angeben wie z.B. `top` oder `left`.

Zugleich ist in STYLE-Attribut des Elementes zu kodieren, ob die Lage des Elementes relativ oder absolut ist:

Bsp. `STYLE="position:absolute;"`

Die Kodierung der neuen Positionen muss im Timer mit mindestens einem der folgenden Attributen erfolgen:

`.begin`
`.by`
`.from`
`.to`

Wert eines jeden dieser Attribute:

String als Liste aus kommagetrennten Werten

Listenelement: Wert, der zum Typ zur Style-Eigenschaft passen muss

Folge der Listenelemente (Wertfolge):

entspricht der Folge der Style-Eigenschaften im Style-Attribut

Bsp.:

`STYLE="position:absolute; top:100px; left:50px; width:100px; height:50px;"`

`TO="200,300"`

also `top` auf 200
`left` auf 300

nicht animiert werden `width` und `height`

Analog gilt das bei Verwendung von `.keySplines`, `.keyTimes` und `.values`

Syntax:

XML `t:ANIMATEMOTION`
 Script `animateMotion`

Beispiel 1:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
function HochZaehlen()
{

```



```

        ID_Div1.innerText = "Zähler: "
                          + (ID_Animation.currTimeState.repeatCount + 1);
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div1"></DIV>
    <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
    >
    </DIV>
    <t:ANIMATEMOTION
        ID="ID_Animation"
        TARGETELEMENT="ID_Div2"
        TO="150,0"
        DUR="1"
        AUTOREVERSE="true"
        REPEATCOUNT="5"
        onrepeat="HochZaehlen()"
    >
    </t:ANIMATEMOTION>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
.div_Klasse{position:absolute; top:195px; height:100px; width:150px; border:solid black 2px;}
</STYLE>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR=".1"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=( ID_Animatemotion.currTimeState.simpleTime);"
    >
        0
    </SPAN>
    <BR>
    <SPAN ID=" ID_Span2"
        CLASS="time_line_klasse"
        DUR=".1"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=( ID_Animatemotion.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <DIV ID="ID_Div"
        CLASSE="div_klasse"
    >
        animierter Div
    </DIV>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
        BEGIN="1; ID_Button.click+1"
        DUR="6s"
        AUTOREVERSE="true"
        CALCMODE="spline"
        KEYSPLINES="0 1 .5 1; .5 1 0 1"
        KEYTIMES="0;.5;1"
        VALUES="25,0;250,50;500,0"
        TARGETELEMENT="ID_Div"
        FILL="hold"
    >
    </t:ANIMATEMOTION>
    <BUTTON ID="ID_Button">Restart</BUTTON>
</BODY>
</HTML>

```



Beispiel 3:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<DIV ID="ID_Div1"
CLASS="time_line_klasse"
STYLE="position:absolute;top:50px;left:50px;height:100px;width:150px;
border:solid black 1px;
"
>
animierter Div mit Vektorgraphik
</DIV>
<t:ANIMATEMOTION ID="ID_Animatemotion1"
TARGETELEMENT="ID_Div1"
BEGIN="ID_Button.click"
PATH="M 0 0 L 100 0 v 100 h -100 V 0"
DUR="3"
>
</:ANIMATEMOTION>
<DIV ID="ID_Div2"
CLASS="time_line_klasse"
STYLE="position:absolute;top:50px;left:250px;height:100px;width:150px;
border:solid black 1px;
"
>
animierter Div mit Vektorgraphik
</DIV>
<t:ANIMATEMOTION ID="ID_Animatemotion2"
TARGETELEMENT="ID_Div2"
BEGIN="ID_Button.click"
PATH="m 0 0 L 100 0 100 100 0 100 z"
DUR="3"
>
</t:ANIMATEMOTION>
<BUTTON ID="ID_Button">Starten/Restarten</BUTTON>
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
window.onload = Rekursion; // ohne () kodieren !!

function Rekursion()
{window.setInterval(Anzeige, 100); }

function Anzeige()
{
ID_Span1.innerHTML = "&nbsp;activeTimeToSegmentTime:&nbsp;";
+ (ID_Animate.activeTimeToSegmentTime(
ID_Div.currTimeState.activeTime
));

ID_Span2.innerHTML = "&nbsp;segmentTime:&nbsp;";
+ (ID_Animate.currTimeState.segmentTime);
}
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Span0"
CLASS="time_line_klasse"
DUR="1"

```



```

        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <DIV ID="ID_Div"
        CLASS="time_line_klasse"
        STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
    >
    </DIV>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
        TARGETELEMENT="ID_Div"
        TO="250,0"
        DUR="3"
        AUTOREVERSE="true"
    >
    </t:ANIMATEMOTION>
    <SPAN ID="ID_Span1">
        activeTimeToSegmentTime:
    </SPAN>
    <BR>
    <SPAN ID="ID_Span2">
        segmentTime:
    </SPAN>
</BODY>
</HTML>

```

Eigenschaften:

.accelerate Beschleunigung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur
 Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
 siehe .decelerate

.accumulate Kumulative Animation eines Elementes auf der Timeline
 Element darf nicht aktiv sein, wenn .accumulate definiert werden soll:
 Element erst danach starten.
 Es kann jede numerische Style-Eigenschaft kumulativ verändert werden und damit
 die kumulative Animation des Elementes in der Style-Eigenschaft erzeugen.
 Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des
 Wertes sein (mathematische Berechnungen erst nach Entfernung der Einheit
 per Stringoperationen möglich)

Name der Eigenschaft laut .attributName
 Änderung des Wertes pro Durchlauf bzw. kumulativ um den Maximalwert laut .by
 Anzahl der Durchläufe bzw. der Wert-Kumulationen laut .repeatCount
 Kumulationsschrittweite laut .calcMode
 Ereignis des Startes eines Durchlaufes bzw. Kumulation um Maximalwert ist onrepeat.
 Wert der Eigenschaft .repeatCount muss > 1 sein
 Kumulation nicht aktiv:
 Nach jedem Durchlauf laut .repeatCount erfolgt Rücksetzen der Style-Eigenschaft des
 Elementes auf den Zustand vor dem Durchlaufbeginn.
 Es wird also die Style-Eigenschaft nicht wertmäßig kumuliert.
 Das Element wird also laut .repeatCount mehrmals animiert.
 Pro Durchlauf wird für die Eigenschaft laut .attributName der Wert
 ab Startwert
 in Schrittweite laut .calcMode
 um den maximalen Wert laut .by
 erhöht.

Kumulation aktiv:
 Die Style-Eigenschaft des Elementes wird pro Durchlauf wertmäßig laut
 .repeatCount um den Wert laut .by kumuliert.
 Kumulationsschrittweite laut .calcMode
 Das Element wird also genau 1 mal animiert.
 Mit Beginn des Startes des Elementes wird für die Eigenschaft laut .attributName
 der Wert ab Startwert
 in Schrittweite laut .calcMode
 um den maximalen Wert aus dem Produkt aus
 .by mal Anzahl der Wiederholungen
 z.B. laut .repeatCount
 erhöht.
 Es wird also über **alle** Wiederholungen der Wert kumuliert.

siehe .additive .calcMode
.additive numerische Werte von Eigenschaften eines Elementes ersetzen oder zu den numerischen Werten der
 gleichnamigen Eigenschaften anderer Elemente addieren
 während der Elemente-Animation auf der Timeline



	<p>Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein gültige Eigenschaften sind z.B. <code>.values</code> oder <code>.by</code> sinnvoll für Kombinationen von Elementen (auch bei Wiederholung eines Elementes) Element darf nicht aktiv sein, wenn <code>.additive</code> definiert werden soll: Element erst danach starten.</p> <p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! siehe <code>.accumulate</code> <code>.by</code> <code>.attributeName</code></p>
<code>.autoReverse</code>	<p>automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline</p>
<code>.begin</code>	<p>Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft <code>.timeAction</code></p>
<code>.by</code>	<p>Wert um den die Werterhöhung bei Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> erfolgen soll</p> <p>Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein wird für Animation in Einheiten zerlegt laut Schrittweite laut <code>.calcMode</code> für die Objekte <code>animate</code>, <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.by</code> wird von <code>.path</code>, <code>.to</code> und <code>.values</code> überschrieben</p> <p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! siehe <code>.attributeName</code> <code>.accumulate</code> <code>.additive</code> <code>.calcMode</code></p>
<code>.calcMode</code>	<p>Schrittweite der Werterhöhung bzw. Art der Animation in der Ebene (1D oder 2D) bei Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code></p>
<code>.decelerate</code>	<p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribut <code>.repeatCount</code> oder <code>.repeatDur</code> Summe der Werte der Attribute <code>.accelerate</code> und <code>.decelerate</code> darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe <code>.accelerate</code></p>
<code>.dur</code>	<p>Dauer Objektaktivitäten laut Eigenschaft <code>.timeAction</code> alternativ: Eigenschaft <code>.end</code></p>
<code>.end</code>	<p>Objektaktivitäten laut Eigenschaft <code>.timeAction</code> beenden ab IE 6.x alternativ: Eigenschaft <code>.dur</code></p>
<code>.fill</code>	<p>Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft <code>endHold</code>, die deprecated ist und nicht mehr verwendet werden darf !!</p>
<code>.from</code>	<p>Startwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> für die Objekte <code>animate</code>, <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.from</code> wird von <code>.path</code> und <code>.values</code> überschrieben</p>
<code>.hasMedia</code>	<p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! Objekt ist HTML-Media-Objekt</p>
<code>.keySplines</code>	<p>Wert eines Pixel-Intervall zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> benötigt <code>.calcMode</code> auf "spline" <code>.keyTimes</code> <code>.values</code> oder <code>.path</code></p>
<code>.keyTimes</code>	<p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! Zeitwert eines Pixel-Intervall zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per <code>.additive</code> oder <code>.accumulate</code> benötigt <code>.calcMode</code> auf "spline" <code>.keySplines</code> <code>.values</code> oder <code>.path</code></p>
<code>.origin</code>	<p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! Bezugspunkt der Animation eines Elementes auf der Timeline nur für Objekt <code>animatemotion</code> (t:ANIMATEMOTION) und <code>.additive</code> muss "replace" sein</p>
<code>.path</code>	<p>Liste von Werten für 2D-Vektorgraphik-Animation eines Elementes auf der Timeline nur für Objekt <code>animatemotion</code> (t:ANIMATEMOTION) siehe <code>.values</code> für 2D-Animation anhand absoluter Koordinaten im Grafiksystem wenn <code>.calcMode</code> auf "paced" so werden Move To-Kommandos ignoriert für die Objekte <code>animate</code>, <code>animateMotion</code> und <code>animateColor</code> gilt: <code>.path</code> wird von keiner Eigenschaft überschrieben <code>.path</code> überschreibt <code>.by</code> <code>.from</code> <code>.to</code> <code>.values</code></p>
<code>.repeatCount</code>	<p>Achtung: Im Objekt <code>animatecolor</code> (t:ANIMATECOLOR) ist <code>.values</code> auch als Farbliste definiert ! aktuelle Nummer der Wiederholung bei Loop</p>
<code>.repeatDur</code>	<p>Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft <code>.dur</code> oder <code>.repeat</code> darf nicht kodiert werden mit der Eigenschaft <code>.repeatCount</code></p>
<code>.restart</code>	<p>generelle Restartmöglichkeit eines Elementes auf der Timeline</p>



	ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.targetElement	ID des zu animierenden Elementes auf der Timeline
	muss für Kindelement immer kodiert werden, wenn nicht das Elternelement animiert werden soll
	muss nicht kodiert werden, wenn kein Elternelement vorliegt
	Achtung: Objekt body ist das oberste Elternelement im Dokument
	und somit haben Elemente innerhalb BODY immer Eltern
	Empfehlung: immer kodieren
	ist der Bezug des Time-Container auf das zu animierende Element
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.to	Endwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline
	per .additive oder .accumulate
	für die Objekte animate, animateMotion und animateColor gilt:
	.to wird von .path und .values überschrieben
	.to überschreibt .by
	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
.values	2D-Animation bei der Elemente-Animation(en) auf der Timeline
	per .additive oder .accumulate
	anhand absoluter Koordinaten im Grafiksystem
	siehe .path für Vektorgraphik-Animation anhand realtiver und absoluter Koordinaten
	Werteliste korrespondiert zu der Werteliste der Eigenschaft .keyTimes
	benötigt .calcMode auf "spline" .keySplines und .keyTimes
	für die Objekte animate, animateMotion und animateColor gilt:
	.values wird von .path überschrieben
	.values überschreibt .by .from .to
	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
Methoden:	
.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren
	identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElementAt()	Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
	also mit Wartezeit ab Beginn der Timeline des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,
	also ohne Wartezeit
	wenn zusätzlich ein weiterer Start (oder mehrere Starts)
	mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
	mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
	also Neustart nach einem erfolgten Start
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen
	identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline
	bzw. die Dauer (Duration) des Elementes abgearbeitet wurde
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.endElementAt()	aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes
	ab Beginn der Timeline stoppen
	sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort,
	also ohne Zeitspanne
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen
	ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
	per Eigenschaft .isActive das Element auf Aktivsein prüfen



<code>.resetElement()</code>	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
<code>.resumeElement()</code>	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode <code>resume()</code> , da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.seekActiveTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft <code>.canSeek</code>
<code>.seekSegmentTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekTo()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekToFrame()</code>	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.segmentTimeToActiveTime()</code>	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
<code>.segmentTimeToSimpleTime()</code>	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
<code>.simpleTimeToSegmentTime()</code>	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

<code>onbegin</code>	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element siehe <code>onend</code> und <code>onrepeat</code>
<code>onend</code>	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code> siehe <code>onbegin</code> und <code>onrepeat</code>
<code>onpause</code>	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
<code>onrepeat</code>	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Heraufreichen des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: onrepeat wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes siehe <code>onend</code> , <code>onbegin</code> , <code>.repeatCount</code> und <code>.repeat</code>
<code>onreset</code>	erzeugt wenn Element zurückgesetzt wurde per Methode <code>.resetElement()</code> , also die Timeline des Elementes den aktuellen Wert laut <code>.begin</code> wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
<code>onresume</code>	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
<code>onreverse</code>	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird



(auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)

.autoReverse muss auf "true" stehen

4.3.2.2.4.3.39.23.9.3.4. *.style.time2.animation Behavior-Objekt des Internet Explorer*

Timerobjekt, das zur Animation eines beliebigen Elementes dient

Media-Element erwartet Datenstrom aus Daten, der meist in einer externen Datei abgelegt ist z.B. Gif-Datei oder Video oder MP3

Spezielle Timerobjekte existieren zusätzlich

z.B. für Audio t:AUDIO
Bild t:IMG,
Media t:MEDIA
Video t:VIDEO
MP3 playItem Objekt in Verbindung mit z.B. t:MEDIA

die direkt auf das Element zugeschnitten sind und mehr Funktionen unterstützen (siehe Objektbeschreibung)

Syntax:

XML t:ANIMATION Element
Script animation

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:IMG ID="ID_Img"
SRC="test.gif"
STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
onmediaerror="alert(' Datei test.gif nicht ladbar !') "
onrepeat="alert(' Aktuelle Wiederholung: ' + ID_Animation.currTimeState.repeatCount);"
>
</t:IMG>
<t:ANIMATION ID="ID_Animation"
TARGETELEMENT="ID_Img"
TO="0,400"
DUR="2"
BEGIN="0"
ACCELERATE="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
>
</t:ANIMATION>
</BODY>
</HTML>
```

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.autoReverse	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
BOUNDARY	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.bufferingProgress	aktueller prozentualer Status des Pufferens eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline Prozentwert des bisher erfolgten Pufferens nur für Media-Datei mit Datenfluss
.canPause	generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline
.canSeek	generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline Auswahl per Seek-Methoden .seekActiveTime()



	.seekSegmentTime()
	.seekTo()
	.seekToFrame()
.clipBegin	nicht jeder Media-Typ unterstützt Seek Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.clipEnd	nicht jeder Media-Typ unterstützt Clipping Endzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.copyright	nicht jeder Media-Typ unterstützt Clipping Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.currentFrame	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Nummer des aktuellen Frame einer Media-Datei auf der Timeline
.decelerate	nicht alle Media-Typen unterstützen Frames Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.downloadCurrent	Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei)
.downloadTotal	nur für Media-Datei mit Datenfluss Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei)
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasAudio	Media-Datei mit Audioinhalt auf der Timeline nicht alle Media-Typen können Audio beinhalten Stummschaltung oder Lautstärkeinstellungen sind dabei egal
.hasDownloadProgres	Start des Donloades einer Media-Datei auf der Timeline
.hasMedia	Objekt ist HTML-Media-Objekt
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline visuelle Daten werden auf dem Bildschirm sichtbar nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls z.B. ist {6BF52A52-394A-11d3-B153-00C04F79FAA6} das ActiveX-Control für den



	Windows Media Player 7.1
.playerObject	Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen siehe Playerbeschreibung des Herstellers vom Player (im Falle von Microsoft sind die Methoden und Eigenschaften im jeweiligen SDK der Player-Version zu finden)
	Playerart laut .player
.rating	Rating der Media-Datei auf der Timeline
.repeatCount	aktuelle Nummer der Wiederholung bei Loop
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe .syncTolerance und .syncmaster
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe .syncTolerance und .syncBehavior
.syncTolerance	zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
TRANSITIONTYPE	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
.type	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software MIME-Typ eines Media-Elementes auf der Timeline
.updateMode	Media-Datei zum Element laut Eigenschaft .src Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline folgende Eigenschaften können geupdatet werden: .autoReverse .begin .dur .end .fill .repeatCount .repeatDur .speed
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt bgsound

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren



	identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.beginElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start
.documentTimeToParentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
.parentTimeToActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren
.parentTimeToDocumentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekSegmentTime()	alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
.seekTo()	siehe Eigenschaft .canSeek aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekToFrame()	nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
.segmentTimeToActiveTime()	nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline



	aktiv wird
	nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element
	erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element
	siehe <code>onend</code> und <code>onrepeat</code>
<code>onend</code>	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird
	erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet
	nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht
	siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code>
	siehe <code>onbegin</code> und <code>onrepeat</code>
<code>onmediacomplete</code>	erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline
	siehe <code>onmediaerror</code>
<code>onmediaerror</code>	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline
	ersetzt das Ereignis <code>onmedialoadfailed</code> , das deprecated ist und nicht mehr verwendet werden darf !
	siehe <code>onmediacomplete</code>
<code>onoutofsync</code>	erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes) sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked": Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch <code>.resetElement()</code>
	siehe <code>onsyncstored</code>
<code>onpause</code>	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
<code>onrepeat</code>	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Heraufreichen des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: onrepeat wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes
	siehe <code>onend</code> , <code>onbegin</code> , <code>.repeatCount</code> und <code>.repeat</code>
<code>onreset</code>	erzeugt wenn Element zurückgesetzt wurde per Methode <code>.resetElement()</code> , also die Timeline des Elementes den aktuellen Wert laut <code>.begin</code> wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
<code>onresume</code>	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
<code>onreverse</code>	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also <code>.repeatCount > 0</code> ist) <code>.autoReverse</code> muss auf "true" stehen
<code>onseek</code>	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: <code>.seekActiveTime()</code> <code>.seekSegmentTime()</code> <code>.seekTo()</code> <code>.seekToFrame()</code>
<code>onsyncstored</code>	erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation siehe <code>onoutofsync</code> sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked"

4.3.2.2.4.3.39.23.9.3.5. *.style.time2.audio Behavior-Objekt des Internet Explorer*

Timerobjekt zur Wiedergabe eines Audio-Elementes im HTML-Dokument

Audio-Element muss nicht mit diesen Objekt wiedergegeben werden, kann aber mit Playlist: siehe Objekt `playItem` und Behavior-Collection `.style.time2.playlist`

Syntax:

XML t:AUDIO
Script audio




```

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=50;"
>50% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=75;"
>75% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=100;"
>100% Volume
</BODY>
</HTML>

```

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.autoReverse	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.Banner	Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playlist ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerAbstract	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playlist ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerMoreInfo	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playlist ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.begin	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
BOUNDARY	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.bufferingProgress	aktueller prozentualer Status des Pufferns eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline Prozentwert des bisher erfolgten Pufferns nur für Media-Datei mit Datenfluss
.canPause	generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline
.canSeek	generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline Auswahl per Seek-Methoden .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame() nicht jeder Media-Typ unterstützt Seek
.clipBegin	Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern



.clipEnd	nicht jeder Media-Typ unterstützt Clipping Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.copyright	nicht jeder Media-Typ unterstützt Clipping Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.currentFrame	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Nummer des aktuellen Frame einer Media-Datei auf der Timeline nicht alle Media-Typen unterstützen Frames
.decelerate	Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.downloadCurrent	Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei) nur für Media-Datei mit Datenfluss
.downloadTotal	Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei)
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasAudio	Media-Datei mit Audioinhalt auf der Timeline nicht alle Media-Typen können Audio beinhalten Stummschaltung oder Lautstärkeinstellungen sind dabei egal
.hasDownloadProgres	Start des Donloades einer Media-Datei auf der Timeline
.hasMedia	Objekt ist HTML-Media-Objekt
.hasPlayList	Verfügbarkeit der Behavior-Collection .style.time2.playlist für Element auf der Timeline, also ob Element eine Playliste hat
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline visuelle Daten werden auf dem Bildschirm sichtbar nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls z.B. ist {6BF52A52-394A-11d3-B153-00C04F79FAA6} das ActiveX-Control für den Windows Media Player 7.1
.playerObject	Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen



	siehe Playerbeschreibung des Herstellers vom Player (im Falle von Microsoft sind die Methoden und Eigenschaften im jeweiligen SDK der Player-Version zu finden)
	Playerart laut .player
.rating	Rating der Media-Datei auf der Timeline
.repeatCount	aktuelle Nummer der Wiederholung bei Loop
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen.
.syncMaster	siehe .syncTolerance und .syncmaster Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !
.syncTolerance	siehe .syncTolerance und .syncBehavior zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
TRANSITIONTYPE	Advanced Stream Redirector (ASX) -Datei: Playlisten-Datei
.type	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software MIME-Typ eines Media-Elementes auf der Timeline Media-Datei zum Element laut Eigenschaft .src
.updateMode	Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline folgende Eigenschaften können geupdatet werden: .autoReverse .begin .dur .end .fill .repeatCount .repeatDur .speed
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt bgsound
Methoden:	
.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.. per Eigenschaft .isActive das Element auf Aktivsein prüfen



<code>.beginElementAt()</code>	<p>Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)</p> <p>also mit Wartezeit ab Beginn der Timeline des Elementes</p> <p>wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,</p> <p>also ohne Wartezeit</p> <p>wenn zusätzlich ein weiterer Start (oder mehrere Starts)</p> <p>mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert</p> <p>mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt</p> <p>also Neustart nach einem erfolgten Start</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.documentTimeToParentTime()</code>	<p>Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline</p> <p>der Eltern des Elementes konvertieren</p>
<code>.endElement()</code>	<p>aktives Element auf der Timeline stoppen</p> <p>identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline</p> <p>bzw. die Dauer (Duration) des Elementes abgearbeitet wurde</p> <p>Alle Kinder des Elementes erhalten Information über Start des Elternelementes</p> <p>und sind somit korrekt auf der Timeline des Elternelementes..</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.endElementAt()</code>	<p>aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes</p> <p>ab Beginn der Timeline stoppen</p> <p>sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes</p> <p>wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort,</p> <p>also ohne Zeitspanne</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.parentTimeToActiveTime()</code>	<p>Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des</p> <p>Elementes konvertieren</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.parentTimeToDocumentTime()</code>	<p>Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert</p> <p>der Timeline des Dokumentes konvertieren</p>
<code>.pauseElement()</code>	<p>aktives Element auf Timeline pausieren lassen</p> <p>ersetzt Methode <code>pause()</code>, die deprecated ist und nicht mehr verwendet werden darf !</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.resetElement()</code>	<p>alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)</p>
<code>.resumeElement()</code>	<p>Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben</p> <p>falls Element nicht pausiert, passiert nichts</p> <p>ersetzt die Methode <code>resume()</code>, da sie deprecated ist und nicht mehr verwendet werden darf</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p>
<code>.seekActiveTime()</code>	<p>aktives Element animieren ab einem Zeitpunkt auf der Timeline</p> <p>wenn Element nicht aktiv, so Fehler erzeugt</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p> <p>alle Media-Typen für Element zulässig</p>
<code>.seekSegmentTime()</code>	<p>siehe Eigenschaft <code>.canSeek</code></p> <p>aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline</p> <p>ohne Wiederholung der Animation</p> <p>wenn Element nicht aktiv, so Fehler erzeugt</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p> <p>Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst</p> <p>nicht alle Media-Typen für Element zulässig</p> <p>wenn unzulässig, so kein Fehler</p> <p>siehe Eigenschaft <code>.canSeek</code></p>
<code>.seekTo()</code>	<p>aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline</p> <p>einschliesslich möglicher Wiederholungen der Animation</p> <p>wenn Element nicht aktiv, so Fehler erzeugt</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p> <p>nicht alle Media-Typen für Element zulässig</p> <p>wenn unzulässig, so kein Fehler</p> <p>siehe Eigenschaft <code>.canSeek</code></p>
<code>.seekToFrame()</code>	<p>Frame eines aktiven Elementes auf der Timeline anwählen</p> <p>wenn Element nicht aktiv, so Fehler erzeugt</p> <p>per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen</p> <p>nicht alle Media-Typen für Element zulässig</p> <p>wenn unzulässig, so kein Fehler</p> <p>siehe Eigenschaft <code>.canSeek</code></p>
<code>.segmentTimeToActiveTime()</code>	<p>Wert der Segment-Timeline des Elements in den korrespondierenden Wert</p> <p>der aktiven Timeline des Elements konvertieren</p>
<code>.segmentTimeToSimpleTime()</code>	<p>Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der</p> <p>Simple-Timeline des Elements konvertieren</p>
<code>.simpleTimeToSegmentTime()</code>	<p>Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der</p> <p>Segment-Timeline des Elementes konvertieren</p>

Events:

<code>onbegin</code>	<p>erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline</p> <p>aktiv wird</p> <p>nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn</p> <p>Eigenschaft <code>.repeatCount > 1</code> ist für das Element</p> <p>erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn</p>
----------------------	--



	Eigenschaft .repeatCount > 1 ist für das Eltern-Element
onend	<p>siehe onend und onrepeat</p> <p>erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount</p> <p>bzw. wenn das aktive Element gestoppt wird</p> <p>erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit des Kindelement ebenfalls endet</p> <p>nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht</p> <p>siehe Methode .endElement() und Eigenschaft .end</p>
onmediacomplete	<p>siehe onbegin und onrepeat</p> <p>erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p>
onmediaerror	<p>siehe onmediaerror</p> <p>erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält</p> <p>für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p> <p>ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf !</p>
onoutofsync	<p>siehe onmediacomplete</p> <p>erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)</p> <p>sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked":</p> <p>Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()</p>
onpause	<p>siehe onsyncstored</p> <p>erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt</p>
onrepeat	<p>erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes</p> <p>nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist</p> <p>nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement)</p> <p>.repeatCount bzw. .repeat muss > 1 sein:</p> <p>onrepeat wird also .repeatCount - 1 mal erzeugt</p> <p>Kind eines Elementes</p>
onreset	<p>siehe onend, onbegin, .repeatCount und .repeat</p> <p>erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(),</p> <p>also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde</p> <p>also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes</p>
onresume	<p>erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt</p>
onreverse	<p>erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird</p> <p>(auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)</p> <p>.autoReverse muss auf "true" stehen</p>
onseek	<p>erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:</p> <p>.seekActiveTime()</p> <p>.seekSegmentTime()</p> <p>.seekTo()</p> <p>.seekToFrame()</p>
onsyncstored	<p>erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird</p> <p>nach vorausgegangenem Abbruch der Synchronisation</p> <p>siehe onoutofsync</p>
ontrackchange	<p>sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked"</p> <p>erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx) in der Playliste gewechselt wurde</p>
onURLFlip	<p>erzeugt wenn ein Skriptkommando, das in einer Advanced Streaming Format (ASF)-Datei (*.asf) liegen, ausgeführt wird</p> <p>Hinweis: window.event.URL Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-Datei von Element auf der Timeline</p> <p>es muss Ereignis onURLFlip aufgetreten sein</p>

4.3.2.2.4.3.39.23.9.3.6. *.style.time2.excl Behavior-Objekt des Internet Explorer*

Timerobjekt für exklusive Animation eines Elementes

Ein Kind in einem EXCL-Time-Container endet sofort, wenn ein anderes Kind mit der Animation startet. Es kann also immer nur 1 Kind animiert werden und das exklusiv gegenüber anderen Kindern.



Wenn ein Kind in einem EXCL-Time-Container ein bereits aktives Kind unterbrechen soll, wobei danach zum unterbrochenen Kind zurückgekehrt werden soll, so ist das Objekt priorityClass zu verwenden.

Syntax:

XML t:EXCL
Script excl

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
  <t:EXCL ID="ID_Excl"
    ENDSYNC="ID_Div2"
  >
    <DIV ID="ID_Div1"
      CLASS="time_line_klasse"
      BEGIN="0"
      DUR="2"
    >
      Zeile 1
    </DIV>
    <DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      BEGIN="2"
      DUR="2"
    >
      Zeile 2
    </DIV>
    <DIV ID="ID_Div3"
      CLASS="time_line_klasse"
      BEGIN="3"
      DUR="2"
    >
      Zeile 3
    </DIV>
  </t:EXCL>
</BODY>
</HTML>
```

Eigenschaften:

.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.autoReverse	automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.decelerate	Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.endSync	Ende der Animation von Elementen in einem gemeinsamen Time-Container bei Ende der Timeline des Eltern-Time-Containers
.repeatDur	Verhalten des Eltern-Time-Containers bezüglich der Animation seiner Kind-Elemente Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen.
.syncTolerance	siehe .syncTolerance und .syncmaster zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster



Methoden:

<code>.activeTimeToParentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.activeTimeToSegmentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.beginElement()</code>	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.. per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.documentTimeToParentTime()</code>	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
<code>.endElement()</code>	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.. per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.parentTimeToActiveTime()</code>	Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.parentTimeToDocumentTime()</code>	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
<code>.resetElement()</code>	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
<code>.seekTo()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekToFrame()</code>	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.segmentTimeToActiveTime()</code>	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
<code>.segmentTimeToSimpleTime()</code>	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
<code>.simpleTimeToSegmentTime()</code>	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

<code>onend</code>	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code>
<code>onmediaerror</code>	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline ersetzt das Ereignis <code>onmedialoadfailed</code> , das deprecated ist und nicht mehr verwendet werden darf ! siehe <code>onmediacomplete</code>
<code>onrepeat</code>	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Heraufreichen des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: onrepeat wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes siehe <code>onend</code> , <code>onbegin</code> , <code>.repeatCount</code> und <code>.repeat</code>



onreset erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(),
 also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht
 und damit zurückgesetzt wurde
 also nicht beim Initialisieren des Elementes und seiner Timeline
 durch Instanziierung des Elementes

onreverse erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird
 (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)
 .autoReverse muss auf "true" stehen

onseek erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:
 .seekActiveTime()
 .seekSegmentTime()
 .seekTo()
 .seekToFrame()

4.3.2.2.4.3.39.23.9.3.7. *.style.time2.img Behavior-Objekt des Internet Explorer*

Timerobjekt, das ein Bild (Image) animiert.

Syntax:

XML t:IMG
 Script img Objekt des Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:IMG ID="ID_Img"
SRC="test.gif"
STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
onmediaerror="alert('Datei test.gif nicht ladbar !')"
onrepeat="alert('Aktuelle Wiederholung: ' + ID_Animation.currTimeState.repeatCount);"
>
</t:IMG>
<t:ANIMATION ID="ID_Animation"
TARGETELEMENT="ID_Img"
TO="0,400"
DUR="2"
BEGIN="0"
ACCELERATE="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
>
</t:ANIMATION>
</BODY>
</HTML>
```

Eigenschaften:

.abstract Beschreibung einer Media-Datei auf der Timeline
 bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT
 vom aktuellen Eintrag geliefert und nicht der Datei selbst

.accelerate Beschleunigung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur
 Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet

.author siehe .decelerate
 Name des Autor der Media-Datei auf der Timeline
 bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven
 Eintrages geliefert und nicht den der Datei.
 Track entspricht playItem Objekt
 Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
 aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

.autoReverse Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
 automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation
 des Elementes auf der Timeline

.begin Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction

BOUNDARY wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software

.bufferingProgress aktueller prozentualer Status des Pufferns eines Datenflusses (Data Stream) einer Media-Datei auf
 der Timeline
 Prozentwert des bisher erfolgten Pufferns
 nur für Media-Datei mit Datenfluss

.canPause generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline



.canSeek	generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline Auswahl per Seek-Methoden .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame() nicht jeder Media-Typ unterstützt Seek
.clipBegin	Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.clipEnd	Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern nicht jeder Media-Typ unterstützt Clipping
.copyright	Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.currentFrame	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Nummer des aktuellen Frame einer Media-Datei auf der Timeline nicht alle Media-Typen unterstützen Frames
.decelerate	Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.downloadCurrent	Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei) nur für Media-Datei mit Datenfluss
.downloadTotal	Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei)
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasAudio	Media-Datei mit Audioinhalt auf der Timeline nicht alle Media-Typen können Audio beinhalten Stummschaltung oder Lautstärkeinstellungen sind dabei egal
.hasDownloadProgres	Start des Donloades einer Media-Datei auf der Timeline
.hasMedia	Objekt ist HTML-Media-Objekt
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline visuelle Daten werden auf dem Bildschirm sichtbar nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline



	<p>Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls z.B. ist {6BF52A52-394A-11d3-B153-00C04F79FAA6} das ActiveX-Control für den Windows Media Player 7.1</p>
.playerObject	<p>Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen siehe Playerbeschreibung des Herstellers vom Player (im Falle von Microsoft sind die Methoden und Eigenschaften im jeweiligen SDK der Player-Version zu finden)</p>
.rating	Playerart laut .player
.repeatCount	Rating der Media-Datei auf der Timeline
.repeatDur	aktuelle Nummer der Wiederholung bei Loop
.restart	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe .syncTolerance und .syncmaster
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe .syncTolerance und .syncBehavior
.syncTolerance	zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
TRANSITIONTYPE	Advanced Stream Redirector (ASX) -Datei: Playlisten-Datei
.type	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software MIME-Typ eines Media-Elementes auf der Timeline Media-Datei zum Element laut Eigenschaft .src
.updateMode	Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline folgende Eigenschaften können geupdatet werden: .autoReverse .begin .dur .end .fill .repeatCount .repeatDur .speed
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt bgsound

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline



	des Segmentes konvertieren
.beginElement()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.beginElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start
.documentTimeToParentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
.parentTimeToActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren
.parentTimeToDocumentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
.resetElement()	per Eigenschaft .isActive das Element auf Aktivsein prüfen alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
.seekActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekSegmentTime()	alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
.seekTo()	siehe Eigenschaft .canSeek aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
.seekToFrame()	siehe Eigenschaft .canSeek Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der



Segment-Timeline des Elementes konvertieren

Events:

onbegin	<p>erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird</p> <p>nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element</p> <p>erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element</p> <p>siehe <code>onend</code> und <code>onrepeat</code></p>
onend	<p>erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code></p> <p>bzw. wenn das aktive Element gestoppt wird</p> <p>erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet</p> <p>nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht</p> <p>siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code></p> <p>siehe <code>onbegin</code> und <code>onrepeat</code></p>
onmediacomplete	<p>erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p> <p>siehe <code>onmediaerror</code></p>
onmediaerror	<p>erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält</p> <p>für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p> <p>ersetzt das Ereignis <code>onmedialoadfailed</code>, das deprecated ist und nicht mehr verwendet werden darf !</p> <p>siehe <code>onmediacomplete</code></p>
onoutofsync	<p>erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)</p> <p>sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked": Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch <code>.resetElement()</code></p> <p>siehe <code>onsyncrestored</code></p>
onpause	<p>erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren</p> <p>auch bei <code>body</code> Objekt</p>
onrepeat	<p>erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes</p> <p>nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist</p> <p>nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Heraufreichen des Ereignisses <code>onrepeat</code> zum Elternelement)</p> <p><code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein:</p> <p><code>onrepeat</code> wird also <code>.repeatCount - 1</code> mal erzeugt</p> <p>Kind eines Elementes</p> <p>siehe <code>onend</code>, <code>onbegin</code>, <code>.repeatCount</code> und <code>.repeat</code></p>
onreset	<p>erzeugt wenn Element zurückgesetzt wurde per Methode <code>.resetElement()</code>, also die Timeline des Elementes den aktuellen Wert laut <code>.begin</code> wieder erreicht und damit zurückgesetzt wurde</p> <p>also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes</p>
onresume	<p>erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird</p> <p>auch für <code>body</code> Objekt</p>
onreverse	<p>erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also <code>.repeatCount > 0</code> ist)</p> <p><code>.autoReverse</code> muss auf "true" stehen</p>
onseek	<p>erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:</p> <p><code>.seekActiveTime()</code></p> <p><code>.seekSegmentTime()</code></p> <p><code>.seekTo()</code></p> <p><code>.seekToFrame()</code></p>
onsyncrestored	<p>erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation</p> <p>siehe <code>onoutofsync</code></p> <p>sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked"</p>

4.3.2.2.4.3.39.23.9.3.8.

.style.time2.media Behavior-Objekt des Internet Explorer

Timerobjekt zur Animation einer Media-Datei

Syntax:

XML t:MEDIA



Script media

Beispiel

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:MEDIA ID="ID_Media"
BEGIN="indefinite;"
SRC="test.wmv"
FILL="remove"
onmediacomplete="Timer2.beginElement();"
>
</t:MEDIA>

<BR>

<BUTTON onclick="ID_Media.beginElement();">Start</BUTTON>
<BUTTON onclick="ID_Media.endElement();">Stop</BUTTON>

<BR>

<B>Volume control:</B>&nbsp;

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='true';"
>Mute

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='false'; ID_Media.volume=25;"
>25% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_Media.mute='false'; ID_Media.volume=50;"
>50% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='false'; ID_Media.volume=75;"
>75% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='false'; ID_Media.volume=100;"
>100% Volume
</BODY>
</HTML>
```

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.autoReverse	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.Banner	Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:



	<p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p> <p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p>
.BannerAbstract	<p>Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList</p> <p>ab IE 6.x</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.BannerMoreInfo	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList</p> <p>ab IE 6.x</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.begin	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction</p>
BOUNDARY	<p>wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software</p>
.bufferingProgress	<p>aktueller prozentualer Status des Pufferens eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline</p> <p>Prozentwert des bisher erfolgten Pufferens</p> <p>nur für Media-Datei mit Datenfluss</p>
.canPause	<p>generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline</p>
.canSeek	<p>generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline</p> <p>Auswahl per Seek-Methoden</p> <p>.seekActiveTime()</p> <p>.seekSegmentTime()</p> <p>.seekTo()</p> <p>.seekToFrame()</p>
.clipBegin	<p>nicht jeder Media-Typ unterstützt Seek</p> <p>Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline</p> <p>Eigenschaft .canSeek muss true liefern</p>
.clipEnd	<p>nicht jeder Media-Typ unterstützt Clipping</p> <p>Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline</p> <p>Eigenschaft .canSeek muss true liefern</p>
.copyright	<p>nicht jeder Media-Typ unterstützt Clipping</p> <p>Copyright der Media-Datei auf der Timeline</p> <p>bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei.</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.currentFrame	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Nummer des aktuellen Frame einer Media-Datei auf der Timeline</p>
.decelerate	<p>nicht alle Media-Typen unterstützen Frames</p> <p>Verlangsamung des Elementes auf der Timeline</p> <p>hat keinen Einfluss auf die Dauer der Timeline</p> <p>auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur</p> <p>Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten</p> <p>wenn ja, so werden beide Attribute ignoriert, also nicht verwendet</p> <p>siehe .accelerate</p>
.downloadCurrent	<p>Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline</p> <p>(Laden des Daten-Stream in Form der Media-Datei)</p> <p>nur für Media-Datei mit Datenfluss</p>
.downloadTotal	<p>Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei auf der Timeline</p> <p>(Laden des Daten-Stream in Form der Media-Datei)</p>
.dur	<p>Dauer Objektaktivitäten laut Eigenschaft .timeAction</p> <p>alternativ: Eigenschaft .end</p>
.end	<p>Objektaktivitäten laut Eigenschaft .timeAction beenden</p> <p>ab IE 6.x</p> <p>alternativ: Eigenschaft .dur</p>
.fill	<p>Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet</p> <p>ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!</p>
.hasAudio	<p>Media-Datei mit Audioinhalt auf der Timeline</p> <p>nicht alle Media-Typen können Audio beinhalten</p> <p>Stummschaltung oder Lautstärkeinstellungen sind dabei egal</p>
.hasDownloadProgres	<p>Start des Donloades einer Media-Datei auf der Timeline</p>



.hasMedia	Objekt ist HTML-Media-Objekt
.hasPlayList	Verfügbarkeit der Behavior-Collection .style.time2.playList für Element auf der Timeline, also ob Element eine Playliste hat
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline visuelle Daten werden auf dem Bildschirm sichtbar nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls z.B. ist {6BF52A52-394A-11d3-B153-00C04F79FAA6} das ActiveX-Control für den Windows Media Player 7.1
.playerObject	Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen siehe Playerbeschreibung des Herstellers vom Player (im Falle von Microsoft sind die Methoden und Eigenschaften im jeweiligen SDK der Player-Version zu finden)
.rating	Playerart laut .player
.repeatCount	Rating der Media-Datei auf der Timeline
.repeatDur	aktuelle Nummer der Wiederholung bei Loop Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe .syncTolerance und .syncmaster
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe .syncTolerance und .syncBehavior
.syncTolerance	zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven



	Eintrages geliefert und nicht den der Datei.
	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
TRANSITIONTYPE	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
.type	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
	MIME-Typ eines Media-Elementes auf der Timeline
	Media-Datei zum Element laut Eigenschaft .src
.updateMode	Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline
	folgende Eigenschaften können geupdatet werden:
	.autoReverse
	.begin
	.dur
	.end
	.fill
	.repeatCount
	.repeatDur
	.speed
.URL	Url aus einem Scriptkommando aus einer Advanced Streaming Format (ASF)-Datei
	es muss Event onURLFlip erzeugt worden sein
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes
	für alle Audio-Elemente des body Objektes auch möglich
	Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus !
	siehe Objekt bgsound

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren
	identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElementAt()	Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
	also mit Wartezeit ab Beginn der Timeline des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,
	also ohne Wartezeit
	wenn zusätzlich ein weiterer Start (oder mehrere Starts)
	mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
	mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
	also Neustart nach einem erfolgten Start
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen
	identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline
	bzw. die Dauer (Duration) des Elementes abgearbeitet wurde
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.endElementAt()	aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes
	ab Beginn der Timeline stoppen
	sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort,
	also ohne Zeitspanne
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.nextTrack()	nächstes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playList
	also Abspielen des Tracks starten
	bzw. nächste Wiederholung starten wenn .repeatCount > 0 und noch nicht
	alle Wiederholungen abgearbeitet wurden
	wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll,
	so wird nicht der erste Track abgespielt, sondern der aktive Track in der
	Wiedergabe gestoppt und danach keine weiteren Aktionen mehr
	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten



<code>.parentTimeToActiveTime()</code>	Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.parentTimeToDocumentTime()</code>	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
<code>.pauseElement()</code>	aktives Element auf Timeline pausieren lassen ersetzt Methode <code>pause()</code> , die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.prevTrack()</code>	vorhergehendes <code>playItem</code> Objekt aktivieren laut Behavior-Collection <code>.style.time2.playList</code> also Abspielen des Tracks starten aber nicht vorhergehende Wiederholung starten wenn <code>.repeatCount > 0</code> und noch nicht alle Wiederholungen abgearbeitet wurden wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll, so wird der erste Track nochmal abgespielt Track entspricht <code>playItem</code> Objekt Trackliste liegt in der Behavior-Collection <code>.style.time2.playList</code> als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei aktiver Track in der Liste: wird abgespielt Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
<code>.resetElement()</code>	
<code>.resumeElement()</code>	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode <code>resume()</code> , da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.seekActiveTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft <code>.canSeek</code>
<code>.seekSegmentTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekTo()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.seekToFrame()</code>	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code>
<code>.segmentTimeToActiveTime()</code>	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
<code>.segmentTimeToSimpleTime()</code>	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
<code>.simpleTimeToSegmentTime()</code>	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren
Events:	
<code>onbegin</code>	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element siehe <code>onend</code> und <code>onrepeat</code>
<code>onend</code>	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit des Kindelement ebenfalls endet nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code>



onmediacomplete	<p>siehe onbegin und onrepeat</p> <p>erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p> <p>siehe onmediaerror</p>
onmediaerror	<p>erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält</p> <p>für Animation per Timeline</p> <p>Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.</p> <p>sinnvoll für Start des Elementes auf der Timeline</p> <p>ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf !</p>
onoutofsync	<p>erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)</p> <p>sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked":</p> <p>Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()</p>
onpause	<p>siehe onsyncrestored</p> <p>erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt</p>
onrepeat	<p>erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes</p> <p>nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist</p> <p>nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement)</p> <p>.repeatCount bzw. .repeat muss > 1 sein:</p> <p>onrepeat wird also .repeatCount - 1 mal erzeugt</p> <p>Kind eines Elementes</p>
onreset	<p>siehe onend, onbegin, .repeatCount und .repeat</p> <p>erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(),</p> <p>also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde</p> <p>also nicht beim Initialisieren des Elementes und seiner Timeline</p> <p>durch Instanziierung des Elementes</p>
onresume	<p>erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird</p> <p>auch für body Objekt</p>
onreverse	<p>erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird</p> <p>(auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)</p> <p>.autoReverse muss auf "true" stehen</p>
onseek	<p>erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:</p> <p>.seekActiveTime()</p> <p>.seekSegmentTime()</p> <p>.seekTo()</p> <p>.seekToFrame()</p>
onsyncrestored	<p>erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird</p> <p>nach vorausgegangenem Abbruch der Synchronisation</p> <p>siehe onoutofsync</p> <p>sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked"</p>
ontrackchange	<p>erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx) in der Playliste gewechselt wurde</p>
onURLFlip	<p>erzeugt wenn ein Scriptkommando, das in einer Advanced Streaming Format (ASF)-Datei (*.asf) liegen, ausgeführt wird</p> <p>Hinweis: window.event.URL Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-Datei von Element auf der Timeline</p> <p>es muss Ereignis onURLFlip aufgetreten sein</p>

4.3.2.2.4.3.39.23.9.3.9. *.style.time2.par Behavior-Objekt des Internet Explorer*

Timer-Container für parallele Animation, wobei sich die Timeline der Kinder überlappen können, aber nicht müssen Standardgemäß beginnen alle Kinder mit Beginn der Eltern-Timeline.

Syntax:

XML t:PAR
Script par

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
```



```

<t:PAR ID="ID_Par"
  BEGIN="0"
  DUR="10"
  FILL="hold"
>
  <t:ANIMATECOLOR ID="ID_Animatecolor1"
    TARGETELEMENT="ID_Div1"
    ATTRIBUTENAME="background-color"
    VALUES="#0000FF;cyan"
    BEGIN="0"
    DUR="5"
    FILL="hold"
  >
  </t:ANIMATECOLOR>
  <t:ANIMATECOLOR ID="ID_Animatecolor2"
    TARGETELEMENT="ID_Div2"
    ATTRIBUTENAME="background-color"
    VALUES="cyan ;#0000FF"
    BEGIN="0"
    DUR="5"
    FILL="hold"
  >
</t:PAR>
<DIV ID="ID_Div1"
  CLASS="time_line_Klasse"
  STYLE="position: absolute; left: 68px; width: 279px; top: 260px; height: 217px;
    border: 1px solid black; background-color: green;
  "
  >
    animierter Div
</DIV>
<DIV ID="ID_Div2"
  CLASS="time_line_classe"
  STYLE="position: absolute; left: 112px; width: 188px; top: 318px; height: 98px;
    padding-left: 3; background-color: gray;
  "
  >
    animierter Div
</DIV>
</BODY>
</HTML>

```

Eigenschaften:

.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.autoReverse	automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.decelerate	Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.endSync	Ende der Animation von Elementen in einem gemeinsamen Time-Container bei Ende der Timeline des Eltern-Time-Containers
.fill	Verhalten des Eltern-Time-Containers bezüglich der Animation seiner Kind-Elemente Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)



<code>.mute</code>	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt <code>bgsound</code>
<code>.repeatCount</code>	aktuelle Nummer der Wiederholung bei Loop
<code>.repeatDur</code>	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft <code>.dur</code> oder <code>.repeat</code> darf nicht kodiert werden mit der Eigenschaft <code>.repeatCount</code>
<code>.restart</code>	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft <code>eventRestart</code> , da diese deprecated ist und nicht mehr verwendet werden darf !!
<code>.speed</code>	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
<code>.syncBehavior</code>	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft <code>.syncmaster</code> kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe <code>.syncTolerance</code> und <code>.syncmaster</code>
<code>.syncTolerance</code>	zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft <code>.syncBehavior</code> auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe <code>.syncBehavior</code> und <code>.syncMaster</code>
<code>.timeAction</code>	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
<code>.timeParent</code>	Zeiger auf das Eltern-Timeline
<code>.volume</code>	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt <code>bgsound</code>

Methoden:

<code>.activeTimeToParentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.activeTimeToSegmentTime()</code>	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.beginElement()</code>	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
<code>.beginElementAt()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
<code>.documentTimeToParentTime()</code>	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
<code>.endElement()</code>	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
<code>.endElementAt()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
<code>.nextTrack()</code>	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nächstes playItem aktivieren laut Behavior-Collection <code>.style.time2.playList</code> also Abspielen des Tracks starten bzw. nächste Wiederholung starten wenn <code>.repeatCount > 0</code> und noch nicht alle Wiederholungen abgearbeitet wurden wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll, so wird nicht der erste Track abgespielt, sondern der aktive Track in der Wiedergabe gestoppt und danach keine weiteren Aktionen mehr Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection <code>.style.time2.playList</code> als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei



	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen
	ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.prevTrack()	vorhergehendes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playList
	also Abspielen des Tracks starten
	aber nicht vorhergehende Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden
	wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll, so wird der erste Track nochmal abgespielt
	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben
	falls Element nicht pausiert, passiert nichts
	ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	alle Media-Typen für Element zulässig
	siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
	ohne Wiederholung der Animation
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
	einschliesslich möglicher Wiederholungen der Animation
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird
	nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element
	erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element
	siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount
	bzw. wenn das aktive Element gestoppt wird
	erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat
	und somit des Kindelement ebenfalls endet
	nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde,



	es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code> siehe <code>onbegin</code> und <code>onrepeat</code>
<code>onmediacomplete</code>	erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline siehe <code>onmediaerror</code>
<code>onmediaerror</code>	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline ersetzt das Ereignis <code>onmedialoadfailed</code> , das deprecated ist und nicht mehr verwendet werden darf ! siehe <code>onmediacomplete</code>
<code>onoutofsync</code>	erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes) sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked": Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch <code>.resetElement()</code>
<code>onpause</code>	siehe <code>onsyncstored</code> erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
<code>onrepeat</code>	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Herausheben des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: <code>onrepeat</code> wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes
<code>onreset</code>	siehe <code>onend</code> , <code>onbegin</code> , <code>.repeatCount</code> und <code>.repeat</code> erzeugt wenn Element zurückgesetzt wurde per Methode <code>.resetElement()</code> , also die Timeline des Elementes den aktuellen Wert laut <code>.begin</code> wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
<code>onresume</code>	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
<code>onreverse</code>	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also <code>.repeatCount > 0</code> ist) <code>.autoReverse</code> muss auf "true" stehen
<code>onseek</code>	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: <code>.seekActiveTime()</code> <code>.seekSegmentTime()</code> <code>.seekTo()</code> <code>.seekToFrame()</code>
<code>onsyncstored</code>	erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation siehe <code>onoutofsync</code> sinnvoll nur bei Zwangssynchronisierung per <code>.syncBehavior</code> auf "locked"

4.3.2.2.4.3.39.23.9.3.10. *.style.time2.playItem Behavior-Objekt des Internet Explorer*

Timerobjekt, das einen Track (Playlisten-Eintrag) repräsentiert

aktiver Track in der Liste: wird gerade abgespielt

Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten

alle Tracks werden per Zeiger in der Behavior-Collection `.style.time2.playlist` gesammelt:

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist,
also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Timerobjekt (Track) kann **nur über** die Behavior-Collection `.style.time2.playlist` referenziert werden

Playliste (Trackliste) stammt aus einer Advanced Stream Redirector (ASX)-Datei, die Playlisteneinträge per Script besitzt

Beispiel für Aufbau eines Eintrages in einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
<entry>
  <title>Testitel</title>
  <author>Testautor</author>
  <copyright>Test 2002</copyright>
```




```

<abstract>WAV Datei</abstract>
<ref href=""></ref>
<banner href = "Testbild.gif" >
    <moreinfo href = "Test.doc"></moreinfo>
    <abstract>besuche www.test.de</abstract>
</banner>
</entry>
</ASX>

```

instanziert in der Behavior-Collection .style.time2.playList als Sammlung der Zeiger aller Tracks (Playlisten-Einträge)
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist,
 also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Syntax:

ein beliebiger Track	zeiger_auf_timer_objekt.playList.item().eigenschaft zeiger_auf_timer_objekt.playList.item().methode
der aktive Track	zeiger_auf_timer_objekt.playList.activeTrack.eigenschaft zeiger_auf_timer_objekt.playList.activeTrack.methode
zeiger_auf_timer_objekt	laut ID-Attribut

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playList.activeTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playList.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playList.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playList.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playList.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playList.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playList.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playList.activeTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
    }
</SCRIPT>
</HEAD>

```



```

<BODY onload="ButtonUpdate()">

    <t:MEDIA          ID="ID_Media"
                      SRC="test.asx"
                      BEGIN="indefinite"
                      TIMEACTION="visibility"
                      onend="ButtonUpdate();"
                      ontrackchange="AnzeigeUpdate();"
                      onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON          ID="ID_Button1"
                    onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON          ID="ID_Button2"
                    onclick="ID_Media.playList.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON          ID="ID_Button3"
                    onclick="ID_Media.playList.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON          ID="ID_Button4"
                    onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>
</HTML>

```

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.begin	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
.copyright	Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playlist: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.dur	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end



.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.hasMedia	Objekt ist HTML-Media-Objekt
.index	Index des Playlisten-Eintrages in der Behavior-Collection .style.time2.playList Diese Eigenschaft ist eigentlich nur sinnvoll für die Kodierung zum aktiven Track..
.rating	Rating der Media-Datei auf der Timeline
.src	Url einer Media-Datei auf der Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeContainer	Typ der Timeline des Objektes
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei MIME-Typ eines Media-Elementes auf der Timeline Media-Datei zum Element laut Eigenschaft .src
.type	
Methoden:	
.setActive()	playItem Objekt als aktiven Track setzen und als solchen in der Behavior-Collection .style.time2.playList registrieren verändert .isActive Track entspricht playItem Objekt (Playlisten-Eintrag) Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei aktiver Track in der Liste: wird abgespielt Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten

4.3.2.2.4.3.39.23.9.3.11. *.style.time2.playList Behavior-Collection des Internet Explorer*

Feld aller playItem Objekte-Referenzen zum Element auf der Timeline

.style.time2.playItem Behavior-Objekt
repräsentiert einen Track (Playlisten-Eintrag)

aktiver Track in der Liste: wird gerade abgespielt

Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten

alle Tracks werden per Zeiger in der Behavior-Collection .style.time2.playList gesammelt:

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist,
also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Timerobjekt (Track) kann **nur über** die Behavior-Collection .style.time2.playList referenziert werden

Playliste (Trackliste) stammt aus einer Advanced Stream Redirector (ASX)-Datei, die Playlisteneinträge per Script besitzt

Beispiel für Aufbau eines Eintrages in einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
<entry>
  <title>Testitel</title>
  <author>Testautor</author>
  <copyright>Test 2002</copyright>
  <abstract>WAV Datei</abstract>
  <ref href=""></ref>
  <banner href = "Testbild.gif" >
    <moreinfo href = "Test.doc"></moreinfo>
    <abstract>besuche www.test.de</abstract>
  </banner>
</entry>
</ASX>
```

instanziert in der Behavior-Collection .style.time2.playList als Sammlung der Zeiger aller Tracks (Playlisten-Einträge)

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist,
also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Syntax:

```
[ var FeldZeiger = ] zeiger_auf_timer_objekt.playList
[ var FeldElementZeiger = ] zeiger_auf_timer_objekt.playList [Index ]
```

Index: Integer und ab 0
muss in [] kodiert sein




```

</t:MEDIA>

<SPAN ID="ID_Span1">Titel:</SPAN>
<BR>
<SPAN ID="ID_Span2">Autor:</SPAN>
<BR>
<SPAN ID="ID_Span3">Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span4">Copyright:</SPAN>
<BR>
<SPAN ID="ID_Span5">Filename:</SPAN>
<BR>
<SPAN ID="ID_Span6">Banner:</SPAN>
<BR>
<SPAN ID="ID_Span7">Banner Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

<BR>
<BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
>
    Start
</BUTTON>
<BUTTON ID="ID_Button2"
        onclick="ID_Media.playList.nextTrack();"
>
    naechster Track
</BUTTON>
<BUTTON ID="ID_Button3"
        onclick="ID_Media.playList.prevTrack();"
>
    vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"
>
    Stop
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    { alert('hasPlayList: ' + ID_Media.hasPlayList); }
</SCRIPT>
</HEAD>
<BODY>
    <t:MEDIA ID="ID_Media"
            SRC="test.wmv"
            BEGIN="0"
            FILL="remove"
            onmediacomplete="Anzeige();"
    >
    </t:MEDIA>
</BODY>
</HTML>

```

Eigenschaften:

.activeTrack

Referenz auf aktives Objekt playItem, das gerade in der Playliste aktiv ist

Achtung: Playliste muss aktiv sein, sonst wird Fehler erzeugt

Aktivstatus prüfen per Eigenschaft .isActive

Track entspricht playItem Objekt

Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

aktiver Track in der Liste: wird abgespielt



.Banner	<p>Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten</p> <p>Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList</p> <p>ab IE 6.x</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.BannerAbstract	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList</p> <p>ab IE 6.x</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.BannerMoreInfo	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList</p> <p>ab IE 6.x</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.dur	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>Dauer Objektaktivitäten laut Eigenschaft .timeAction</p> <p>alternativ: Eigenschaft .end</p>
.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
Methoden:	
.item()	<p>Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern</p> <p>außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!</p>
.nextTrack()	<p>nächstes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playList</p> <p>also Abspielen des Tracks starten</p> <p>bzw. nächste Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden</p> <p>wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll, so wird nicht der erste Track abgespielt, sondern der aktive Track in der Wiedergabe gestoppt und danach keine weiteren Aktionen mehr</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
.prevTrack()	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei</p> <p>aktiver Track in der Liste: wird abgespielt</p> <p>Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten</p> <p>vorhergehendes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playList</p> <p>also Abspielen des Tracks starten</p> <p>aber nicht vorhergehende Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden</p> <p>wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll, so wird der erste Track nochmal abgespielt</p> <p>Track entspricht playItem Objekt</p> <p>Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:</p> <p>Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
4.3.2.2.4.3.39.23.9.3.12.	<i>.style.time2.priorityClass Behavior-Objekt des Internet Explorer</i>
Timerobjekt	<p>als Kind vom Time-Container t:EXCL</p> <p>definiert das Animations-Verhalten mehrerer Timerobjekte t:PRIORITYCLASS</p> <p>oder aller Elemente innerhalb eines gemeinsamen t:PRIORITYCLASS</p> <p>im Time-Container t:EXCL</p>
Time-Container t:EXCL	<p>jedes Kind ein t:PRIORITYCLASS</p> <p>alle Kinder sequentiell kodieren: Verschachtelung von t:PRIORITYCLASS nicht zulässig ist</p> <p>Kodierungsfolge ist die absteigende Prioritätsfolge für die Animation.</p> <p>Das erste kodierte t:PRIORITYCLASS hat die höchste Priorität</p> <p>Das letzte kodierte t:PRIORITYCLASS hat die niedrigste Priorität.</p>



Im ersten kodierten t:PRIORITYCLASS darf kein Klassen-Attribut HIGHER kodiert sein, dafür aber LOWER möglich. Soll LOWER kodiert werden, so muss es mindestens 1 Nachfolger-Kind geben.

Im letzten kodierten t:PRIORITYCLASS darf kein Klassen-Attribut LOWER kodiert sein, dafür aber HIGHER möglich. Soll HIGHER kodiert werden, so muss es mindestens 1 Vorgänger-Kind geben.

Es gibt **genau ein** t:PRIORITYCLASS für die Kodierung des Klassen-Attributes PEERS.

Syntax:

XML <t:PRIORITYCLASS klassen_attribut = kette >
Script priorityClass

klassen_attribut Gross-Klein egal

HIGHER Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation

eines höher priorisierten Kindes

Kette String

Es gilt folgende Ausgangssituation:

Kind X hat kodiertes HIGHER animiert bereits

Kind Y hat höhere Priorität als Kind X animiert nicht

"pause" Default wenn Kind Y startet, pausiert

Kind X solange, bis Kind Y fertig ist

"stop" wenn Kind Y startet, bricht Kind X ab

kein Abbruch von **bereits** pausierenden Kindern

LOWER Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines niedriger priorisierten Kindes

Kette String

Es gilt folgende Ausgangssituation:

Kind X hat niedrigere Priorität als Kind Y animiert nicht

Kind Y hat LOWER kodiert animiert bereits

"defer" Default wenn Kind X startet, pausiert es sofort

und wartet, bis Kind Y fertig ist

"never" Kind X startet nie

PEERS Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines Kindes

Kette String

Es gilt folgende Ausgangssituation:

Kind X animiert nicht
Kind Y animiert bereits

"pause" wenn Kind X startet, pausiert Kind Y solange, bis Kind X fertig ist

"stop" Default wenn Kind X startet, bricht Kind Y ab

kein Abbruch von **bereits** pausierenden Kindern



"defer"	wenn Kind X startet, dann pausiert es sofort und solange, bis Kind Y fertig ist
"never"	Kind X startet nie
kein	Abbruch von bereits pausierenden Kindern

Beispiel 1:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="0;indefinite;"
>
<t:PRIORITYCLASS ID="ID_Prior1">
<SPAN ID="ID_Span1"
CLASS="time_line_Klasse"
BEGIN="5"
DUR="5"
>
Test1
</SPAN>
</t:PRIORITYCLASS>
<t:PRIORITYCLASS ID="ID_Prior2" HIGHER="stop">
<SPAN ID="ID_Span2"
CLASS="time_line_Klasse"
BEGIN="0"
DUR="10"
>
Test2
</SPAN>
</t:PRIORITYCLASS>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
</BUTTON>
</BODY>
</HTML>
```

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="0;indefinite;"
>
<t:PRIORITYCLASS ID="ID_Prior1">
<SPAN ID="ID_Span1"
CLASS="time_line_Klasse"
BEGIN="5"
DUR="5"
>
Test1
</SPAN>
</t:PRIORITYCLASS>
<t:PRIORITYCLASS ID="ID_Prior2" LOWER="never">
<SPAN ID="ID_Span2"
CLASS="time_line_Klasse"
BEGIN="0"
DUR="10"
>
```



```

>
    Test2
</SPAN>
<t:/PRIORITYCLASS>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
</BUTTON>
</BODY>
</HTML>

```

Beispiel 3:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
    BEGIN="0;indefinite;"
>
    <t:PRIORITYCLASS ID="ID_Prior" PEERS="pause">
        <SPAN ID="ID_Span1"
            CLASS="time_line_Klasse"
            BEGIN="0"
            DUR="10"
        >
            Test1
        </SPAN>
        <SPAN ID="ID_Span2"
            CLASS="time_line_Klasse"
            BEGIN="5"
            DUR="3"
        >
            Test2
        </SPAN>
    </t:PRIORITYCLASS>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
</BUTTON>
</BODY>
</HTML>

```

Eigenschaften:

.higher

Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines höher priorisierten Kindes
 Time-Container ist Behavior-Objektes .style.time2.excl
 Folge der in HTML kodierten Objekte t:PRIORITYCLASS entspricht der absteigenden Folge der Prioritäten der Animation.
 Jedes Kind ist ein t:PRIORITYCLASS

darf **kein** weiteres Behavior-Objekt .style.time2.priorityClass besitzen
 wenn mehrere pausierende Kinder existieren:

Das zuletzt pausierte Kind startet zuerst.
 Das zuerst pausierte Kind startet zuletzt.

.lower

Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines niedriger priorisierten Kindes
 Time-Container ist Behavior-Objektes .style.time2.excl
 Folge der in HTML kodierten Objekte t:PRIORITYCLASS entspricht der absteigenden Folge der Prioritäten der Animation.
 Jedes Kind ist ein t:PRIORITYCLASS

darf **kein** weiteres Behavior-Objekt .style.time2.priorityClass besitzen
 wenn mehrere pausierende Kinder existieren:

Das zuletzt pausierte Kind startet zuerst.
 Das zuerst pausierte Kind startet zuletzt.

.peers

Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines Kindes
 Time-Container ist Behavior-Objektes .style.time2.excl
 Es gibt **genau** ein t:PRIORITYCLASS, in dem **kein** weiteres Behavior-Objekt .style.time2.priorityClass kodiert sein darf.

wenn mehrere pausierende Kinder existieren:
 Das zuletzt pausierte Kind startet zuerst.
 Das zuerst pausierte Kind startet zuletzt.
 siehe Objekt currTimeState und Behavior .style.time2



Methoden:

keine

4.3.2.2.4.3.39.23.9.3.13. *.style.time2.ref Behavior-Objekt des Internet Explorer*

Timerobjekt für Referenz innerhalb des Dokumentes

Syntax:

XML	t:REF
Script	ref

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.autoReverse	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.Banner	Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerAbstract	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerMoreInfo	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.begin	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
BOUNDARY	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.bufferingProgress	aktueller prozentualer Status des Pufferns eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline Prozentwert des bisher erfolgten Pufferns nur für Media-Datei mit Datenfluss
.canPause	generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline
.canSeek	generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline Auswahl per Seek-Methoden .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame() nicht jeder Media-Typ unterstützt Seek
.clipBegin	Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern nicht jeder Media-Typ unterstützt Clipping
.clipEnd	Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern nicht jeder Media-Typ unterstützt Clipping
.copyright	Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei.



	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
.currentFrame	Nummer des aktuellen Frame einer Media-Datei auf der Timeline
	nicht alle Media-Typen unterstützen Frames
.decelerate	Verlangsamung des Elementes auf der Timeline
	hat keinen Einfluss auf die Dauer der Timeline
	auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur
	Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten
	wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
	siehe .accelerate
.downloadCurrent	Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline
	(Laden des Daten-Stream in Form der Media-Datei)
	nur für Media-Datei mit Datenfluss
.downloadTotal	Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei
	auf der Timeline
	(Laden des Daten-Stream in Form der Media-Datei)
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction
	alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
	ab IE 6.x
	alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber
	bevor die Timeline des Elternelementes endet
	ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasAudio	Media-Datei mit Audioinhalt auf der Timeline
	nicht alle Media-Typen können Audio beinhalten
	Stummschaltung oder Lautstärkeinstellungen sind dabei egal
.hasDownloadProgres	Start des Donloades einer Media-Datei auf der Timeline
.hasMedia	Objekt ist HTML-Media-Objekt
.hasPlayList	Verfügbarkeit der Behavior-Collection .style.time2.playlist für Element auf der Timeline,
	also ob Element eine Playliste hat
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline
	visuelle Daten werden auf dem Bildschirm sichtbar
	nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline
	nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline
	nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline
	Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline
	Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet
	(ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline
	Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet
	(ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline
	wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen
	beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern
	siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline
	Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss
	ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls
	z.B. ist {22d6f312-b0f6-11d0-94ab-0080c74c7e95} das ActiveX-Control für den
	Windows Media-Player im Internet Explorer.
.playerObject	Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt
	besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist
	Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen
	siehe Playerbeschreibung des Herstellers vom Player
	(im Falle von Microsoft sind die Methoden und Eigenschaften
	im jeweiligen SDK der Player-Version zu finden)
	Playerart laut .player
.rating	Rating der Media-Datei auf der Timeline
.repeatCount	aktuelle Nummer der Wiederholung bei Loop
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline



	verlangt kodierte Eigenschaft .dur oder .repeat
	darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline
	ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten)
	Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen.
	siehe .syncTolerance und .syncmaster
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit).
	nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior)
	ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !
	siehe .syncTolerance und .syncBehavior
.syncTolerance	zeitliche Toleranz für Zwangs-Synchronisation von Elementen auf der Timeline also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist
	sinnvoll vorallem bei Time-Container (Gruppen von Objekten)
	siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline
	Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline
	bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei.
	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
TRANSITIONTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.type	MIME-Typ eines Media-Elementes auf der Timeline
	Media-Datei zum Element laut Eigenschaft .src
.updateMode	Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline
	folgende Eigenschaften können geupdatet werden:
	.autoReverse
	.begin
	.dur
	.end
	.fill
	.repeatCount
	.repeatDur
	.speed
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich
	Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus !
	siehe Objekt bgsound

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren
	identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElementAt()	Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
	also mit Wartezeit ab Beginn der Timeline des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit
	wenn zusätzlich ein weiterer Start (oder mehrere Starts)
	mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
	mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt



	also Neustart nach einem erfolgten Start
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
.nextTrack()	per Eigenschaft .isActive das Element auf Aktivsein prüfen nächstes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playlist also Abspielen des Tracks starten bzw. nächste Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll, so wird nicht der erste Track abgespielt, sondern der aktive Track in der Wiedergabe gestoppt und danach keine weiteren Aktionen mehr Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei aktiver Track in der Liste: wird abgespielt Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren
.parentTimeToDocumentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
.prevTrack()	per Eigenschaft .isActive das Element auf Aktivsein prüfen vorhergehendes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playlist also Abspielen des Tracks starten aber nicht vorhergehende Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll, so wird der erste Track nochmal abgespielt Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist. Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei aktiver Track in der Liste: wird abgespielt Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
.seekActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig



	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen
	wenn Element nicht aktiv, so Fehler erzeugt
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.setActive()	playItem Objekt als aktiven Track setzen und als solchen in der Behavior-Collection .style.time2.playlist registrieren
	verändert .isActive
	Track entspricht playItem Objekt (Playlisten-Eintrag)
	Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
	Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird
	nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element
	erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element
	siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount
	bzw. wenn das aktive Element gestoppt wird
	erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet
	nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht
	siehe Methode .endElement() und Eigenschaft .end
	siehe onbegin und onrepeat
onmediacomplete	erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline
	Media-Element benötigt vor seiner Animation den Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.
	sinnvoll für Start des Elementes auf der Timeline
	siehe onmediaerror
onmediaerror	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält
	für Animation per Timeline
	Media-Element benötigt vor seiner Animation den Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.
	sinnvoll für Start des Elementes auf der Timeline
	ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf !
	siehe onmediacomplete
onoutofsync	erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)
	sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked":
	Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()
	siehe onsyncstored
onpause	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren
	auch bei body Objekt
onrepeat	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes
	nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist
	nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement)
	.repeatCount bzw. .repeat muss > 1 sein:
	onrepeat wird also .repeatCount - 1 mal erzeugt
	Kind eines Elementes
	siehe onend, onbegin, .repeatCount und .repeat



onreset	erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(), also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanzierung des Elementes
onresume	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
onreverse	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist) .autoReverse muss auf "true" stehen
onseek	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame()
onsyncrestored	erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation siehe onoutofsync
ontrackchange	sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked" erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx) in der Playliste gewechselt wurde

4.3.2.2.4.3.39.23.9.3.14. *.style.time2.set Behavior-Objekt des Internet Explorer*

Timerobjekt zur Animation eines HTML-Elementes anhand eines Styles unter Verwendung der Attribute **ATTRIBUTENAME** und **TARGETELEMENT**

Syntax:

XML	t:SET
Script	set

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SET ID="ID_Set"
ATTRIBUTENAME="width"
BEGIN="2"
TO="300"
DUR="5"
TARGETELEMENT="ID_Div"
>
</t:SET>
<DIV ID="ID_Div"
STYLE="width:100px; height:50px;"
>
Test
</DIV>
</BODY>
</HTML>
```

Eigenschaften:

.attributeName	Bezeichner einer Style-Eigenschaft, also Name eines Attributes, für die Animation anhand dieses Attributes auf der Timeline
.autoReverse	siehe .accumulate .by .additive .calcMode automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasMedia	Objekt ist HTML-Media-Objekt
.repeatDur	Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!



.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.targetElement	ID des zu animierenden Elementes auf der Timeline muss für Kindelement immer kodiert werden, wenn nicht das Elternelement animiert werden soll muss nicht kodiert werden, wenn kein Elternelement vorliegt Achtung: Objekt body ist das oberste Elternelement im Dokument und somit haben Elemente innerhalb BODY immer Eltern Empfehlung: immer kodieren ist der Bezug des Time-Container auf das zu animierende Element
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.to	Endwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline per .additive oder .accumulate für die Objekte animate, animateMotion und animateColor gilt: .to wird von .path und .values überschrieben .to überschreibt .by Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
Methoden:	
.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.beginElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start
.documentTimeToParentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
.parentTimeToActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren
.parentTimeToDocumentTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation



	<p>wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code></p>
<code>.seekTo()</code>	<p>aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code></p>
<code>.seekToFrame()</code>	<p>Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler siehe Eigenschaft <code>.canSeek</code></p>
<code>.segmentTimeToActiveTime()</code>	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
<code>.segmentTimeToSimpleTime()</code>	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
<code>.setActive()</code>	<p>playItem Objekt als aktiven Track setzen und als solchen in der Behavior-Collection <code>.style.time2.playlist</code> registrieren verändert <code>.isActive</code> Track entspricht playItem Objekt (Playlisten-Eintrag) Trackliste liegt in der Behavior-Collection <code>.style.time2.playlist</code> als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.</p>
<code>.simpleTimeToSegmentTime()</code>	<p>Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei aktiver Track in der Liste: wird abgespielt Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren</p>
Events:	
<code>onbegin</code>	<p>erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element siehe <code>onend</code> und <code>onrepeat</code></p>
<code>onend</code>	<p>erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code> siehe <code>onbegin</code> und <code>onrepeat</code></p>
<code>onpause</code>	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
<code>onrepeat</code>	<p>erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Heraufreichen des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: onrepeat wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes siehe <code>onend</code>, <code>onbegin</code>, <code>.repeatCount</code> und <code>.repeat</code></p>
<code>onreset</code>	<p>erzeugt wenn Element zurückgesetzt wurde per Methode <code>.resetElement()</code>, also die Timeline des Elementes den aktuellen Wert laut <code>.begin</code> wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes</p>
<code>onresume</code>	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
<code>onreverse</code>	<p>erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also <code>.repeatCount > 0</code> ist) <code>.autoReverse</code> muss auf "true" stehen</p>



4.3.2.2.4.3.39.23.9.3.15. .style.time2.seq Behavior-Objekt des Internet Explorer

Timerobjekt zur sequentiellen Animation

Syntax:

XML	t:SEQ
Script	seq

Beispiel 1:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:SEQ ID="ID_Seq"
        REPEATDUR="27"
        BEGIN="0"
    >
        <DIV ID="ID_Div1"
            CLASS="time_line_klasse"
            DUR="3"
        >
            3 Sekunden lang anzeigen
        </DIV>
        <DIV ID="ID_Div2"
            CLASS="time_line_klasse"
            DUR="4"
        >
            4 Sekunden lang anzeigen
        </DIV>
    </t:SEQ>
</BODY>
</HTML>
```

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <DIV STYLE="height:100px">
        <t:SEQ ID="ID_Seq" REPEATCOUNT="indefinite" >
            <t:MEDIA ID="ID_Media1"
                SRC = "test1.jpg"
                STYLE="position:absolute;"
                DUR="3"
                TIMECONTAINER="par"
                FILL="transition"
            >
                <t:TRANSITIONFILTER ID="ID_Transfilter1"
                    TYPE="fade"
                    DUR="2"
                >
                </t:TRANSITIONFILTER>
            </t:MEDIA>
            <t:MEDIA ID="ID_Media2"
                SRC = "test2.jpg"
                STYLE="position:absolute;"
                DUR="3"
                TIMECONTAINER="par"
                FILL="transition"
            >
                <t:TRANSITIONFILTER ID="ID_Transfilter2"
                    TYPE="ClockWipe"
                    DUR="2"
                >
                </t:TRANSITIONFILTER>
            </t:MEDIA>
        </t:SEQ>
```




```

</DIV>
</BODY>
</HTML>

```

Beispiel 3:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SEQ ID="ID_Seq"
CLASS="time_line_klasse"
>
<SPAN ID="ID_Span1"
CLASS="time_line_klasse"
TIMEACTION="display"
DUR="1"
>
5...
</SPAN>
<SPAN ID="ID_Span2"
CLASS="time_line_klasse"
TIMEACTION="display"
DUR="1"
>
4...
</SPAN>
<BR>
<IMG ID="ID_Img"
SRC="test.gif"
CLASS="time_line_klasse"
DUR="indefinite"
>
</t:SEQ>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Seq.beginElement();">
Start der Timeline
</BUTTON>
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SEQ ID="ID_Seq"
BEGIN="indefinite;"
>
<DIV ID="ID_Div1"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:15px;left:25px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div2"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:30px;left:50px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div3"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:35px;left:75px;width:100px;height:100px;"
>

```



```

</DIV>
<DIV ID="ID_Div4"
      CLASS="time_line_klasse"
      DUR="10"
      style="position:relative;top:50px;left:100px;width:100px;height:100px;"
    >
</DIV>
</t:SEQ>
<BR>
<BUTTON ID="ID_Button1" onclick="ID_Seq.beginElement();"> start</BUTTON>
<BUTTON ID="ID_Button2" onclick="ID_Seq.nextElement();">next </BUTTON>
<BUTTON ID="ID_Button3" onclick="ID_Seq.prevElement();">previous</BUTTON>
<BUTTON ID="ID_Button4" onclick="ID_Seq.endElement();">stop</BUTTON>
</BODY>
</HTML>

```

Eigenschaften:

.accelerate	<p>Beschleunigung des Elementes auf der Timeline</p> <p>hat keinen Einfluss auf die Dauer der Timeline</p> <p>auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur</p> <p>Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten</p> <p>wenn ja, so werden beide Attribute ignoriert, also nicht verwendet</p> <p>siehe .decelerate</p>
.autoReverse	<p>automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation</p> <p>des Elementes auf der Timeline</p>
.begin	<p>Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction</p>
.decelerate	<p>Verlangsamung des Elementes auf der Timeline</p> <p>hat keinen Einfluss auf die Dauer der Timeline</p> <p>auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur</p> <p>Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten</p> <p>wenn ja, so werden beide Attribute ignoriert, also nicht verwendet</p> <p>siehe .accelerate</p>
.dur	<p>Dauer Objektaktivitäten laut Eigenschaft .timeAction</p> <p>alternativ: Eigenschaft .end</p>
.end	<p>Objektaktivitäten laut Eigenschaft .timeAction beenden</p> <p>ab IE 6.x</p> <p>alternativ: Eigenschaft .dur</p>
.fill	<p>Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber</p> <p>bevor die Timeline des Elternelementes endet</p> <p>ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!</p>
.hasMedia	<p>Objekt ist HTML-Media-Objekt</p>
.mediaDur	<p>Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline</p>
.mediaHeight	<p>Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein</p> <p>aktuelle Höhe des Media-Elementes in Pixels auf der Timeline</p> <p>Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet</p> <p>(ansonsten STYLE-Angabe zu .style.height im Element kodieren)</p>
.mediaWidth	<p>aktuelle Breite des Media-Elementes in Pixels auf der Timeline</p> <p>Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet</p> <p>(ansonsten STYLE-Angabe zu .style.width im Element kodieren)</p>
.mute	<p>Audio aktiv oder aus (stumm) auf der Timeline</p> <p>wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen</p> <p>beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern</p> <p>siehe Objekt bgsound</p>
.repeatCount	<p>aktuelle Nummer der Wiederholung bei Loop</p>
.repeatDur	<p>Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline</p> <p>verlangt kodierte Eigenschaft .dur oder .repeat</p> <p>darf nicht kodiert werden mit der Eigenschaft .repeatCount</p>
.restart	<p>generelle Restartmöglichkeit eines Elementes auf der Timeline</p> <p>ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!</p>
.speed	<p>aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)</p>
.syncBehavior	<p>Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes,</p> <p>sinnvoll vorallem bei Time-Container (Gruppen von Objekten)</p> <p>Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen.</p> <p>siehe .syncTolerance und .syncmaster</p>
.syncTolerance	<p>zeitliche Toleranz für Zwangs-Synchronisation von Elementen auf der Timeline</p> <p>also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist</p> <p>sinnvoll vorallem bei Time-Container (Gruppen von Objekten)</p> <p>siehe .syncBehavior und .syncMaster</p>
.systemBitrate	<p>wird hier nicht erklärt</p>
.systemCaptions	<p>wird hier nicht erklärt</p>
.systemLanguage	<p>Sprache festlegen für das Objekt</p>
.systemOverdubOrSubtitle	<p>wird hier nicht erklärt</p>
.timeAction	<p>Aktion des Objektes in der Timeline</p>



Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
 .timeContainer Typ der Timeline des Objektes
 .timeParent Zeiger auf das Eltern-Timeline
 .volume Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich
 Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus !
 siehe Objekt bgsound

Methoden:

.activeTimeToParentTime() Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .activeTimeToSegmentTime() Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .beginElement() Element auf der Timeline starten, also aktivieren
 identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
 Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .beginElementAt() Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
 also mit Wartezeit ab Beginn der Timeline des Elementes
 wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort, also ohne Wartezeit
 wenn zusätzlich ein weiterer Start (oder mehrere Starts)
 mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
 mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
 also Neustart nach einem erfolgten Start
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .documentTimeToParentTime() Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
 .endElement() aktives Element auf der Timeline stoppen
 identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde
 Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes..
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .endElementAt() aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen
 sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
 wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort, also ohne Zeitspanne
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .nextElement() nächstes Element aus aktiver t:SEQ animieren auf der Timeline
 wenn kein nächstes Element, so kein Fehler erzeugt
 .nextTrack() nächstes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playlist
 also Abspielen des Tracks starten
 bzw. nächste Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden
 wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll, so wird nicht der erste Track abgespielt, sondern der aktive Track in der Wiedergabe gestoppt und danach keine weiteren Aktionen mehr
 Track entspricht playItem Objekt
 Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
 aktiver Track in der Liste: wird abgespielt
 Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
 .parentTimeToActiveTime() Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .parentTimeToDocumentTime() Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
 .pauseElement() aktives Element auf Timeline pausieren lassen
 ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 .prevElement() vorhergehendes Element aus aktiver t:SEQ animieren auf der Timeline
 wenn kein vorhergehendes Element, so wird Timeline von t:SEQ auf 0 gesetzt
 .prevTrack() vorhergehendes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playlist
 also Abspielen des Tracks starten
 aber **nicht** vorhergehende Wiederholung starten wenn .repeatCount > 0 und noch nicht alle Wiederholungen abgearbeitet wurden
 wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll, so wird der erste Track nochmal abgespielt



	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
.seekActiveTime()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	alle Media-Typen für Element zulässig siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.setActive()	playItem Objekt als aktiven Track setzen und als solchen in der Behavior-Collection .style.time2.playlist registrieren verändert .isActive Track entspricht playItem Objekt (Playlisten-Eintrag)
	Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
	aktiver Track in der Liste: wird abgespielt
	Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren
Events:	
onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element
onend	siehe onend und onrepeat erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode .endElement() und Eigenschaft .end
onmediacomplete	siehe onbegin und onrepeat erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline Media-Element benötigt vor seiner Animation den Datenfluss des Mediums aus einer



	Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline siehe onmediaerror
onmediaerror	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation den Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf ! siehe onmediacomplete
onoutofsync	erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes) sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked": Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement() siehe onsyncrestored
onpause	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
onrepeat	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement) .repeatCount bzw. .repeat muss > 1 sein: onrepeat wird also .repeatCount - 1 mal erzeugt Kind eines Elementes siehe onend, onbegin, .repeatCount und .repeat
onreset	erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(), also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
onresume	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
onreverse	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist) .autoReverse muss auf "true" stehen
onseek	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame()
onsyncrestored	erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation siehe onoutofsync sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked"

4.3.2.2.4.3.39.23.9.3.16. *.style.time2.switch Behavior-Objekt des Internet Explorer*

Timerobjekt für automatische Auswahl und Aktivierung mindestens eines Elementes aus einer Menge von Elementen

Auswahl nur möglich anhand genau eines der folgenden systemnahen Attribute:

.systemBitrate
.systemCaptions
.systemLanguage
.systemOverdubOrSubtitle

das die Elemente unterstützen **und kodiert** haben **müssen**.

anhand der aktuellen zutreffenden Systemeinstellung:

Wenn der Attributwert mit der aktuellen Systemeinstellung übereinstimmt, so erfolgt die Auswahl und damit die
Abarbeitung **nur** des ausgewählten Elementes.

wird nicht angezeigt

nur von Elementen, die auf der gemeinsamen Hierarchie-Ebene **direkt unter** der von t:SWITCH kodiert wurden (Auswahl-
Ebene): Nur in der Auswahl-Ebene kodierte Attribute obiger Art werden für die Auswahl verwendet.

Element nur auswählbar, wenn auf der Hierarchie-Ebene **direkt unter** der von t:SWITCH kodiert wurde (Auswahl-Ebene):

Wenn Element ein HTML-Container ist, also HTML-Kinder hat, so werden in den Kindern eventuell kodierte Attribute
von obiger Art **nicht** in den Vergleich der Auswahl einbezogen, da die Kinder in einer tieferen Hierarchie-Ebene liegen
und nicht in der Auswahl-Ebene.

Aktivierung entspricht Abarbeitung des Elementes je nach Art des Elementes

Werden **nicht** obige Attribute unterstützt bzw. kodiert, so wird das Element **immer ausgewählt** (auch HTML-Kommentar),
also abgearbeitet !

wenn mit Attribut .timeContainer, also Element ist selbst ein Time-Container, so wird dieses Attribut ignoriert, wenn das Element
nicht automatisch ausgewählt wurde



Syntax:

XML t:SWITCH
Script switch

Beispiel für Auswahl anhand der aktuellen Sprache des Usersystems per Attribut **SYSTEMLANGUAGE**:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:SWITCH ID="ID_Switch">

        <t:SEQ    ID="ID_Seq"
                STYLE="font-size:18pt;color:#ff0000"
        >

            <DIV    ID="ID_Div1"
                CLASS="time_line_klasse"
                DUR="1.5"

            >
                Falls keine Sprachbezeichnung sichtbar,

            </DIV>
            <DIV    ID="ID_Div2"
                CLASS="time_line_klasse"
                DUR="1.5"

            >
                so konnte nichts ausgewaehlt werden !

            </DIV>
        </t:SEQ>
        <BR>

        <SPAN    ID="Span1"
                CLASS="time_line_Klasse"
                SYSTEMLANGUAGE="es"

        >
            Espanol
        </SPAN>

        <SPAN    ID="Span2"
                CLASS="time_line_Klasse"
                SYSTEMLANGUAGE="en"

        >
            English
        </SPAN>

        <MARQUEE    ID="ID_Marquee"
                SYSTEMLANGUAGE="de"
                LOOP=1
                HEIGHT=200
                WIDTH=200
                SCROLLAMOUNT=10
                SCROLLDELAY=20
                BEHAVIOR="SLIDE"
                DIRECTION="DOWN"
                STYLE="position:absolute; top:0; left:10"

        >
            Deutsch
        </MARQUEE>
        <BR>
        <SPAN    ID="Span3"
                CLASS="time_line_Klasse"
                SYSTEMLANGUAGE="de"

        >
            wurde ausgewaehlt.
        </SPAN>

    </t:SWITCH>
</BODY>
</HTML>
```

Hinweise: Die Auswahl erfolgt anhand von Span1, Span2, MARQUEE und Span3 durch Vergleich auf die



aktuelle Systemsprache mit Spanisch (es) bzw. Englisch (en) bzw. Deutsch (de).
Ist das Usersystem deutschsprachig, so werden MARQUEE und Span3 aktiviert.
t:SEQ wird immer aktiviert, da kein SYSTEMLANGUAGE kodiert wurde.

Eigenschaften:

.hasMedia	Objekt ist HTML-Media-Objekt
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt

Methoden:

keine

Events:

keine

4.3.2.2.4.3.39.23.9.3.17. .style.time2.transitionFilter Behavior-Objekt des Internet Explorer

Timerobjekt für **visuelle** Filteranimation als "Übergang"

zwischen 2 aktiven Elementen:

z.B. zwischen 2 Bilder, also Übergang vom Quell-Bild zum Ziel-Bild, wobei beide Bilder eine gemeinsame Position besitzen:

vor dem Start der Animation ist **mindestens** das Quell-Element sichtbar
nach dem Ende der Animation ist **mindestens** das Ziel-Element sichtbar

z.B. zwischen DIV und einem Bild, dass im DIV animiert wird: Bild taucht allmählich im DIV auf.

vor dem Start der Animation ist DIV ohne **oder** schon mit teilweiseem Bild
nach dem Ende der Animation ist DIV mit (auch teilweise) Bild

eines aktiven Elementes:

z.B. DIV, der sich wandelt (komplett oder teilweise)

Übergang:	von sichtbar zu unsichtbar:	vor Start des Filters sichtbar (auch teilweise) nach Start des Filters unsichtbar (auch teilweise)
oder	von unsichtbar zu sichtbar:	vor Start des Filters unsichtbar (auch teilweise) nach Start des Filters sichtbar (auch teilweise)

Element(e) müssen sichtbar sein bzw. sichtbar machbar sein, also aktiv sein.

Animation auch per Attribut TARGETELEMENT anhand des ID-Attributes des zu animierenden Elementes möglich.

Ein Element kann mit mehreren sequentiell kodierten transitionsFilter animiert werden, die z.B. innerhalb eines gemeinsamen Eltern-Time-Containers
oder von BODY
liegen.

Das Schreiben der transitionFilter-spezifischen Eigenschaften .by. .from .to und .value darf nur erfolgen, wenn der Übergangsfiler **nicht** aktiv ist. Also **erst** Filter deaktivieren, dann Eigenschaften belegen, **dann** Filter starten.

Bsp.:

```
<SCRIPT>
    object.endElement();
    object.by='0.4';
    object.beginElement();
</SCRIPT>
```

siehe auch Objekt filter

Syntax:

XML	t:TRANSITIONFILTER
Script	transitionFilter

Beispiel 1 Bild-Übergang innerhalb eines DIV:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:TRANSITIONFILTER ID="ID_Transfilter"
                        TYPE="fade"
                        DUR="8"
                        END="4"
                        TARGETELEMENT="ID_Div"
    >
</ t:TRANSITIONFILTER>
<DIV ID="ID_Div">
```




```

        CLASS="time_line_klasse"
        DUR="indefinite"
        STYLE="background-image:url(test.gif); background-repeat: no-repeat;"
    >
    </DIV>
</BODY>
</HTML>

```

Beispiel 2 für einen DIV, der sich wandelt:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <DIV ID="ID_Div"
        CLASS="time_line_klasse"
        DUR="8"
        STYLE="width:270"
    >
        <t:TRANSITIONFILTER ID="ID_Transfilter1"
            BEGIN="ID_Div.begin"
            DUR="3"
            TYPE="clockWipe"
        >
        </t:TRANSITIONFILTER>

        <t:TRANSITIONFILTER ID="ID_Transfilter2"
            BEGIN="ID_Div.end - 3"
            DUR="3"
            MODE="out"
            TYPE="fade"
        >
        </t:TRANSITIONFILTER>

        Test
    </DIV>
</BODY>
</HTML>

```

Beispiel 3 für Animation eines Elementes mit mehreren transitionFilter, die in einem gemeinsamen Eltern-Container liegen

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <DIV ID="ID_Div" STYLE="height:100px">
        <t:SEQ ID="ID_Seq"
            REPEATCOUNT="indefinite"
        >
            <t:MEDIA ID="ID_Media1"
                SRC = "test1.jpg"
                STYLE="position:absolute; ....."
                DUR="3"
                TIMECONTAINER="par"
                FILL="transition"
            >
                <t:TRANSITIONFILTER ID="ID_Transfilter1"
                    TYPE="fade"
                    DUR="2"
                >
                </t:TRANSITIONFILTER>

                <t:TRANSITIONFILTER ID="ID_Transfilter2"
                    TYPE="ClockWipe"
                    DUR="2"
                >
                </t:TRANSITIONFILTER>
            </t:MEDIA>
        </t:SEQ>
    </DIV>

```




```

</t:TRANSITIONFILTER>

</t:MEDIA>
<t:MEDIA          ID="ID_Media2"
                  SRC = "test2.jpg"
                  STYLE="position:absolute; ...."
                  DUR="3"
                  TIMECONTAINER="par"
                  FILL="transition"

>
    <t:TRANSITIONFILTER  ID="ID_Transfilter3"
                        TYPE="ClockWipe"
                        DUR="2"

    >
    </t:TRANSITIONFILTER>
    <t:TRANSITIONFILTER  ID="ID_Transfilter4"
                        TYPE="fade"
                        DUR="2"

    >
    </t:TRANSITIONFILTER>

</t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <DIV ID="ID_Div" STYLE="height:100px">
    <t:SEQ  ID="ID_Seq"
        REPEATCOUNT="indefinite"

    >
        <t:MEDIA          ID="ID_Media1"
                        SRC = "test1.jpg"
                        ALT="Test1"
                        STYLE="position:absolute; ...."
                        DUR="3"
                        TIMECONTAINER="par"
                        FILL="transition"

        >
            <t:TRANSITIONFILTER  ID="ID_Transfilter1"
                                TYPE="fade"
                                DUR="2"

            >
            </t:TRANSITIONFILTER>

            <t:TRANSITIONFILTER  ID="ID_Transfilter2"
                                TYPE="ClockWipe"
                                DUR="2"

            >
            </t:TRANSITIONFILTER>

        <MARQUEE          ID="ID_Marquee"
                        CLASS="time_line_klasse"
                        TIMECONTAINER="seq"
                        REPEATCOUNT="indefinite"
                        STYLE="position:absolute; ...."

        >
            <IMG          ID="ID_Img1"
                        SRC="test2.gif"
                        CLASS="time"
                        DUR="4"
                        ALT="Test2"

            >
            <IMG          ID="ID_Img2"
                        SRC="test3.gif"

```



```

        CLASS="time"
        DUR="4"
        ALT="Test3"
    >
    <IMG ID="ID_Img3"
        SRC="test4.gif"
        CLASS="time"
        DUR="4"
        ALT="Test4"
    >
    <IMG ID="ID_Img4"
        SRC="test5.gif"
        CLASS="time"
        DUR="4"
        ALT="Test5"
    >
    </MARQUEE>
</t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>

```

Beispiel 5 für 2 DIV, die sich unabhängig voneinander wandeln:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:TRANSITIONFILTER ID="ID_Transfilter1"
        TYPE="barWipe"
        DUR="3"
        TARGETELEMENT="ID_Div1"
        MODE="in"
    >
    </t:TRANSITIONFILTER>

    <t:TRANSITIONFILTER ID="ID_Transfilter2"
        TYPE="barWipe"
        DUR="3"
        TARGETELEMENT="ID_Div2"
        MODE="out"
    >
    </t:TRANSITIONFILTER>

    <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="indefinite"
        STYLE= "position:relative; left:20px; width:420px; height:100px;
            background-image:url(test.gif); background-repeat: no-repeat;
            "
    >
    </DIV>

    <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="indefinite"
        STYLE= "position:relative; left:20px; width:420px; height:100px;
            background-image:url(test.gif); background-repeat: no-repeat;
            "
    >
    </DIV>
</BODY>
</HTML>

```

Beispiel 6 für mehrere transitionFilter innerhalb Body, die ein gemeinsames Element animieren:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">

```



```

<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter1"
TYPE="barWipe"
SUBTYPE="leftToRight"
DUR="3"
TARGETELEMENT="ID_Div1"
>
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter2"
TYPE="starWipe"
SUBTYPE="fivePoint"
DUR="3"
TARGETELEMENT="ID_Div1"
>
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter3"
TYPE="barnDoorWipe"
DUR="3"
TARGETELEMENT="ID_Div2"
FROM="0"
TO="1"
CALCMODE="linear"
MODE="in"
>
</t:TRANSITIONFILTER>

<DIV ID="ID_Div0" STYLE="height:170px;">

<DIV ID="ID_Div1"
CLASS="time_line_klasse"
STYLE="position:absolute; top:150px; left:20px; background-color:#3366CC;
padding:10px; height:80; color:white;
"
>
Test1
</DIV>

<DIV ID="ID_Div2"
CLASS="time_line_klasse"
STYLE="position:absolute; top:185px; left:60px; background-color:#FFCC00;
padding:10px; height:80;
"
>
Test2
</DIV>
</DIV>
</BODY>
</HTML>

```

Beispiel 7 für Start der Animation durch Ereignis onclick aus Klick auf ein Button:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

<t:TRANSITIONFILTER ID="ID_Transfilter1"
BEGIN="ID_Div1.begin"
TYPE="barWipe"
DUR="5"
TARGETELEMENT="ID_Div1"
VALUES="1;17;27;37;47;56;65;71;82;92;1.0"

```



```

        CALCMODE="discrete"
    >
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter2"
    BEGIN="ID_Div1.begin"
    TYPE="barWipe"
    DUR="5"
    TARGETELEMENT="ID_Div2"
    VALUES=".1;.17;.27;.37;.47;.56;.65;.71;.82;.92;1.0"
    CALCMODE="linear"
>
</t:TRANSITIONFILTER>
<BR>
<INPUT TYPE="button" ID="ID_Button" VALUE="Start Transition">

<DIV ID="ID_Div1"
    CLASS="time_line_Klasse"
    BEGIN="ID_Button.click"
    DUR="indefinite"
    STYLE="position:relative; left:20px; width:420px; height:100px;
        background-image:url(test.gif); background-repeat: no-repeat;
    "
>
</DIV>

<DIV ID="ID_Div2"
    CLASS="time_line_Klasse"
    BEGIN="ID_Button.click"
    DUR="indefinite"
    STYLE="position:relative; left:20px; width:420px; height:100px;
        background-image:url(test.gif); background-repeat: no-repeat;
    "
>
</DIV>
</BODY>
</HTML>

```

Eigenschaften:

Das Schreiben der transitionFilter-spezifischen Eigenschaften `.by`, `.from`, `.to` und `.value` darf nur erfolgen, wenn der Übergangsfiler **nicht** aktiv ist. Also **erst** Filter deaktivieren, dann Eigenschaften belegen, **dann** Filter starten.

Bsp.:

```

<SCRIPT>
    object.endElement();
    object.by=0.4;
    object.beginElement();
</SCRIPT>

```

.accelerate Beschleunigung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute `.repeatCount` oder `.repeatDur`
 Summe der Werte der Attribute `.accelerate` und `.decelerate` darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
 siehe `.decelerate`

.accumulate Kumulative Animation eines Elementes auf der Timeline
 Element darf nicht aktiv sein, wenn `.accumulate` definiert werden soll:
 Element erst danach starten.
 Es kann jede numerische Style-Eigenschaft kumulativ verändert werden und damit
 die kumulative Animation des Elementes in der Style-Eigenschaft erzeugen.
 Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des
 Wertes sein (mathematische Berechnungen erst nach Entfernung der Einheit
 per Stringoperationen möglich)
 Name der Eigenschaft laut `.attributName`
 Änderung des Wertes pro Durchlauf bzw. kumulativ um den Maximalwert laut `.by`
 Anzahl der Durchläufe bzw. der Wert-Kumulationen laut `.repeatCount`
 Kumulationsschrittweite laut `.calcMode`
 Ereignis des Startes eines Durchlaufes bzw. Kumulation um Maximalwert ist `onrepeat`.
 Wert der Eigenschaft `.repeatCount` muss > 1 sein
 Kumulation nicht aktiv:
 Nach jedem Durchlauf laut `.repeatCount` erfolgt Rücksetzen der Style-Eigenschaft des
 Elementes auf den Zustand vor dem Durchlaufbeginn.
 Es wird also die Style-Eigenschaft nicht wertmäßig kumuliert.
 Das Element wird also laut `.repeatCount` mehrmals animiert.
 Pro Durchlauf wird für die Eigenschaft laut `.attributName` der Wert



	ab Startwert in Schrittweite laut .calcMode um den maximalen Wert laut .by
	erhöht.
	Kumulation aktiv: Die Style-Eigenschaft des Elementes wird pro Durchlauf wertmäßig laut .repeatCount um den Wert laut .by kumuliert. Kumulationsschrittweite laut .calcMode Das Element wird also genau 1 mal animiert. Mit Beginn des Startes des Elementes wird für die Eigenschaft laut .attributeName der Wert ab Startwert in Schrittweite laut .calcMode um den maximalen Wert aus dem Produkt aus .by mal Anzahl der Wiederholungen z.B. laut .repeatCount
	erhöht. Es wird also über alle Wiederholungen der Wert kumuliert.
.additive	siehe .additive .calcMode numerische Werte von Eigenschaften eines Elementes ersetzen oder zu den numerischen Werten der gleichnamigen Eigenschaften anderer Elemente addieren während der Elemente-Animation auf der Timeline Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein gültige Eigenschaften sind z.B. .values oder .by sinnvoll für Kombinationen von Elementen (auch bei Wiederholung eines Elementes) Element darf nicht aktiv sein, wenn .additive definiert werden soll: Element erst danach starten. Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! siehe .accumulate .by .attributeName
.autoReverse	automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.begin .by	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction Schrittweite des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt als Offset zum Wert der kodierten Eigenschaft .from des Behavior Schrittweite als Prozentsatz des Überganges nach dessen Vollendung nicht zusammen mit Eigenschaft .to oder .value kodieren, sonst wird .by ignoriert
.calcMode	Interpolation der Aufteilung der internen Animationsschritte auf Schrittweite laut .by Schritt laut .from bei optionalen .to Gesamtschritt bei fehlendem .by bzw. .from .to Animationsschritte laut .values per Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt Bsp.: Lineare Aufteilung durch Interpolation: Interne Schritte zu gleichen Teilen verteilt keine Interpolation, so Animation laut Schrittweite, also eventuell nicht fließende Animation Größe des internen Animationsschrittes: Je nach der Gesamtdauer der Animation laut .dur des Behavior Verlangsamung des Elementes auf der Timeline
.decelerate	hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.dur .end	zeitliche Gesamtdauer des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.from	Start-Prozentsatz des kompletten Überganges bei Start der Animation des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt Start des Übergangsfilters mit bereits teilweise vollendetem Übergang nicht zusammen mit Eigenschaft .value kodieren, sonst wird .from ignoriert
.keySpline	Wert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per .additive oder .accumulate benötigt .calcMode auf "spline" .keyTimes .values oder .path
.keyTimes	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! Zeitwert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline per .additive oder .accumulate benötigt .calcMode auf "spline" .keySpline .values oder .path
.mode	Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert ! Ergebnis der Animation des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt, also Sichtbarkeit bzw. Unsichtbarkeit nach Vollendung des Filters
.repeatCount .repeatDur	aktuelle Nummer der Wiederholung bei Loop Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline



	verlangt kodierte Eigenschaft .dur oder .repeat
	darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline
	ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.subtype	visuelle Erscheinungsform des Überganges per .style.time2.transitionFilter Behavior-Objekt,
	also visuelle Art und Weise des Überganges
	optional kodierbar zur Eigenschaft .type des Behavior
.targetElement	ID des zu animierenden Elementes auf der Timeline
	muss für Kindelement immer kodiert werden, wenn nicht das Elternelement animiert werden soll
	muss nicht kodiert werden, wenn kein Elternelement vorliegt
	Achtung: Objekt body ist das oberste Elternelement im Dokument
	und somit haben Elemente innerhalb BODY immer Eltern
	Empfehlung: immer kodieren
	ist der Bezug des Time-Container auf das zu animierende Element
.to	Ende-Prozentsatz des kompletten Überganges bei Ende der Animation des Übergangsfilters
	per .style.time2.transitionFilter Behavior-Objekt
	Ende des Übergangsfilters mit nur teilweise vollendetem Übergang
	nicht zusammen mit Eigenschaften .by und .value kodieren, sonst wird .to ignoriert
.type	visuelle Erscheinungsform des Überganges per .style.time2.transitionFilter Behavior-Objekt,
	also visuelle Art und Weise des Überganges
	optional kodierbar ist zusätzlich die Eigenschaft .subtype des Behavior
.values	Animationsschritte als Prozentsatz des kompletten Überganges
	per .style.time2.transitionFilter Behavior-Objekt
	nicht zusammen mit Eigenschaften .by und .to und .from kodieren, da diese sonst ignoriert werden

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren
	identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes..
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElementAt()	Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
	also mit Wartezeit ab Beginn der Timeline des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,
	also ohne Wartezeit
	wenn zusätzlich ein weiterer Start (oder mehrere Starts)
	mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
	mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
	also Neustart nach einem erfolgten Start
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen
	identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline
	bzw. die Dauer (Duration) des Elementes abgearbeitet wurde
	Alle Kinder des Elementes erhalten Information über Start des Elternelementes
	und sind somit korrekt auf der Timeline des Elternelementes.
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.endElementAt()	aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes
	ab Beginn der Timeline stoppen
	sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
	wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort,
	also ohne Zeitspanne
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktiven Timeline des Elementes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen
	ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben
	falls Element nicht pausiert, passiert nichts
	ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline
	wenn Element nicht aktiv, so Fehler erzeugt



	per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen
	alle Media-Typen für Element zulässig
	siehe Eigenschaft <code>.canSeek</code>
<code>.seekSegmentTime()</code>	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
<code>.seekTo()</code>	siehe Eigenschaft <code>.canSeek</code> aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen nicht alle Media-Typen für Element zulässig wenn unzulässig, so kein Fehler
<code>.segmentTimeToActiveTime()</code>	siehe Eigenschaft <code>.canSeek</code> Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
<code>.segmentTimeToSimpleTime()</code>	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
<code>.simpleTimeToSegmentTime()</code>	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren
Events:	
<code>onbegin</code>	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft <code>.repeatCount > 1</code> ist für das Eltern-Element siehe <code>onend</code> und <code>onrepeat</code>
<code>onend</code>	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut <code>.repeatCount</code> bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn <code>.fill</code> mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode <code>.endElement()</code> und Eigenschaft <code>.end</code> siehe <code>onbegin</code> und <code>onrepeat</code>
<code>onpause</code>	erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
<code>onrepeat</code>	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe <code>onbegin</code>), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für <code>onrepeat</code> kodiert hat (kein Herausheben des Ereignisses <code>onrepeat</code> zum Elternelement) <code>.repeatCount</code> bzw. <code>.repeat</code> muss > 1 sein: <code>onrepeat</code> wird also <code>.repeatCount - 1</code> mal erzeugt Kind eines Elementes siehe <code>onend</code> , <code>onbegin</code> , <code>.repeatCount</code> und <code>.repeat</code>
<code>onresume</code>	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
<code>onreverse</code>	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also <code>.repeatCount > 0</code> ist) <code>.autoReverse</code> muss auf "true" stehen
<code>onseek</code>	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: <code>.seekActiveTime()</code> <code>.seekSegmentTime()</code> <code>.seekTo()</code> <code>.seekToFrame()</code>

4.3.2.2.4.3.39.23.9.3.18. *.style.time2.video Behavior-Objekt des Internet Explorer*

Timerobjekt zur Wiedergabe von Video

Syntax:

```
XML      t:VIDEO
Script   video
```

Beispiel 1:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
```



```

        .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<SCRIPT>
    function Aendern()
    {
        ID_Video.updateMode = ID_Select1.options.value;
        ID_Video.speed      = ID_Select1.options.value;
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                     SRC="test.avi"
                     UPDATEMODE="reset"
                     STYLE="width:175px; height:150px;"
    >
    </t:VIDEO>
    <BR>
    updateMode waehlen:
    <SELECT NAME="ID_Select1">
        <OPTION VALUE="auto">Auto</OPTION>
        <OPTION VALUE="reset" SELECTED>Reset</OPTION>
    </SELECT>
    <BR>
    Geschwindigkeit waehlen:
    <SELECT NAME="ID_Select2">
        <OPTION VALUE="0.25">25%</OPTION>
        <OPTION VALUE="0.50">50%</OPTION>
        <OPTION VALUE="0.75">75%</OPTION>
        <OPTION VALUE="1" SELECTED>100% </OPTION>
        <OPTION VALUE="2" SELECTED>200% </OPTION>
    </SELECT>
    <BR>
    <BUTTON ID="ID_Button1" onClick="Aendern();">
        Geschwindigkeit aendern
    </BUTTON>
    <BUTTON ID="ID_Button1" onClick="document.body.beginElement()">
        Restart
    </BUTTON>
</CENTER>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        switch (ID_Video.currTimeState.state)
        {
            case 1:
                if (ID_Video.currTimeState.isPaused == true)
                {
                    ID_Button2.disabled = true;
                    ID_Button3.disabled = true;
                    ID_Button4.disabled = false;
                    ID_Button5.disabled = false;
                }
                else
                {
                    ID_Button2.disabled = true;
                    ID_Button3.disabled = false;
                    ID_Button4.disabled = true;
                    ID_Button5.disabled = false;
                }
                break;
            case 0:
                ID_Button2.disabled = false;

```




```

        ID_Button3.disabled = true;
        ID_Button4.disabled = true;
        ID_Button5.disabled = true;
        break;
    case 4:
        ID_Button2.disabled = false;
        ID_Button3.disabled = true;
        ID_Button4.disabled = true;
        ID_Button5.disabled = true;
        break;
    }
}
</SCRIPT>
<SCRIPT FOR="document" EVENT="onclick" LANGUAGE="JScript">
    Anzeige();
</SCRIPT>
</HEAD>
<BODY onload=" Anzeige()">
    <t:VIDEO ID="ID_Video"
        CLASS="time_line_klasse"
        BEGIN="indefinite"
        STYLE="position:absolute;top:90px;height:150px;"
        FILL="remove"
        SRC="/test /media/movie.avi"
    >
    </ t:VIDEO>
    <SPAN ID="ID_Span"
        STYLE="position:absolute;top:255px;"
    >
        Status: 0
    </SPAN>
    <P STYLE="position:absolute;top:280px;">
        <BUTTON ID="ID_Button1"
            onclick=
                "ID_Span.innerText='Status: ' + ID_Video.currTimeState.state"
        >
            Aktueller Status
        </BUTTON>
        <BUTTON ID="ID_Button2"
            onclick="ID_Video.beginElement();"
        >
            Beginn
        </BUTTON>
        <BUTTON ID="ID_Button3"
            onclick="ID_Video.pauseElement();"
        >
            Pause
        </BUTTON>
        <BUTTON ID="ID_Button4"
            onclick="ID_Video.resumeElement();"
        >
            Resume
        </BUTTON>
        <BUTTON ID="ID_Button5"
            onclick=
                "ID_Video.endElement();ID_Span.innerText='Status: 0'"
        >
            Stop
        </BUTTON>
    </P>
</BODY>
</HTML>

```

Eigenschaften:

.abstract	Beschreibung einer Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT vom aktuellen Eintrag geliefert und nicht der Datei selbst
.accelerate	Beschleunigung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .decelerate
.author	Name des Autor der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven



	Eintrages geliefert und nicht den der Datei.
	Track entspricht playItem Objekt
	Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.autoReverse	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation des Elementes auf der Timeline
.Banner	Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerAbstract	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.BannerMoreInfo	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList ab IE 6.x Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.begin	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
BOUNDARY	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
.bufferingProgress	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software aktueller prozentualer Status des Pufferns eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline Prozentwert des bisher erfolgten Pufferns nur für Media-Datei mit Datenfluss
.canPause	generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline
.canSeek	generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline Auswahl per Seek-Methoden .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame()
.clipBegin	nicht jeder Media-Typ unterstützt Seek Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.clipEnd	nicht jeder Media-Typ unterstützt Clipping Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline Eigenschaft .canSeek muss true liefern
.copyright	nicht jeder Media-Typ unterstützt Clipping Copyright der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven Eintrages geliefert und nicht den der Datei.
	Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
.currentFrame	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei Nummer des aktuellen Frame einer Media-Datei auf der Timeline
.decelerate	nicht alle Media-Typen unterstützen Frames Verlangsamung des Elementes auf der Timeline hat keinen Einfluss auf die Dauer der Timeline auch wirksam bei Wiederholungen per Attribute .repeatCount oder .repeatDur Summe der Werte der Attribute .accelerate und .decelerate darf nicht 1 überschreiten wenn ja, so werden beide Attribute ignoriert, also nicht verwendet siehe .accelerate
.downloadCurrent	Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei) nur für Media-Datei mit Datenfluss
.downloadTotal	Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei auf der Timeline (Laden des Daten-Stream in Form der Media-Datei)



.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur
.fill	Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber bevor die Timeline des Elternelementes endet ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet werden darf !!
.hasAudio	Media-Datei mit Audioinhalt auf der Timeline nicht alle Media-Typen können Audio beinhalten Stummschaltung oder Lautstärkeinstellungen sind dabei egal
.hasDownloadProgres	Start des Donloades einer Media-Datei auf der Timeline
.hasMedia	Objekt ist HTML-Media-Objekt
.hasPlayList	Verfügbarkeit der Behavior-Collection .style.time2.playList für Element auf der Timeline, also ob Element eine Playliste hat
.hasVisual	Media-Datei mit visuellem Inhalt auf der Timeline visuelle Daten werden auf dem Bildschirm sichtbar nicht alle Media-Typen können sichtbare Daten beinhalten
IMMEDIATEEND	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.isStreamed	Media-Datei mit Datenfluss (Data Stream) auf der Timeline nicht alle Media-Typen unterstützen Datenstrom
.latestMediaTime	Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline nicht für Media-Datei mit Datenfluss
LONGTRANSITION	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mediaDur	Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
.mediaHeight	aktuelle Höhe des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
.mediaWidth	aktuelle Breite des Media-Elementes in Pixels auf der Timeline Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
.mimeType	MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
MODULATE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
MOTIFNAME	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.mute	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt bgsound
.player	Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls z.B. ist {22d6f312-b0f6-11d0-94ab-0080c74c7e95} das ActiveX-Control für den Windows Media-Player im Internet Explorer.
.playerObject	Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen siehe Playerbeschreibung des Herstellers vom Player (im Falle von Microsoft sind die Methoden und Eigenschaften im jeweiligen SDK der Player-Version zu finden)
.rating	Playerart laut .player
.repeatCount	Rating der Media-Datei auf der Timeline
.repeatDur	aktuelle Nummer der Wiederholung bei Loop Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline verlangt kodierte Eigenschaft .dur oder .repeat darf nicht kodiert werden mit der Eigenschaft .repeatCount
.restart	generelle Restartmöglichkeit eines Elementes auf der Timeline ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.speed	aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
.src	Url einer Media-Datei auf der Timeline
.syncBehavior	Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vorallem bei Time-Container (Gruppen von Objekten) Es muss die Eigenschaft .syncmaster kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen. siehe .syncTolerance und .syncmaster
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe .syncTolerance und .syncBehavior
.syncTolerance	zeitliche Toleranz für Zwangs-Synchronisation von Elementen auf der Timeline



	also wenn Eigenschaft .syncBehavior auf "locked" gesetzt ist sinnvoll vorallem bei Time-Container (Gruppen von Objekten) siehe .syncBehavior und .syncMaster
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
.timeContainer	Typ der Timeline des Objektes
.timeParent	Zeiger auf das Eltern-Timeline
.title	Titel der Media-Datei auf der Timeline bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven Eintrages geliefert und nicht den der Datei. Track entspricht playItem Objekt Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste: Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
TRANSITIONTYPE	Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
.type	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software MIME-Typ eines Media-Elementes auf der Timeline Media-Datei zum Element laut Eigenschaft .src
.updateMode	Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline folgende Eigenschaften können geupdatet werden: .autoReverse .begin .dur .end .fill .repeatCount .repeatDur .speed
.URL	Url aus einem Scriptkommando aus einer Advanced Streaming Format (ASF)-Datei es muss Event onURLFlip erzeugt worden sein
.volume	Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus ! siehe Objekt bgsound

Methoden:

.activeTimeToParentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline der Eltern konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.activeTimeToSegmentTime()	Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren per Eigenschaft .isActive das Element auf Aktivsein prüfen
.beginElement()	Element auf der Timeline starten, also aktivieren identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.
.beginElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) also mit Wartezeit ab Beginn der Timeline des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht har, so startet des Kindelement sofort, also ohne Wartezeit wenn zusätzlich ein weiterer Start (oder mehrere Starts) mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt also Neustart nach einem erfolgten Start per Eigenschaft .isActive das Element auf Aktivsein prüfen
.documentTimeToParentTime()	Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
.endElement()	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.
.endElementAt()	per Eigenschaft .isActive das Element auf Aktivsein prüfen aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes wenn das Elternelement diesen Zeitpunkt schon erreicht har, so endet des Kindelement sofort, also ohne Zeitspanne per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToActiveTime()	Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des



	Elementes konvertieren
	per Eigenschaft .isActive das Element auf Aktivsein prüfen
.parentTimeToDocumentTime()	Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
.pauseElement()	aktives Element auf Timeline pausieren lassen ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf ! per Eigenschaft .isActive das Element auf Aktivsein prüfen
.resetElement()	alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
.resumeElement()	Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben falls Element nicht pausiert, passiert nichts ersetzt die Methode resume(), da sie deprecated ist und nicht mehr verwendet werden darf per Eigenschaft .isActive das Element auf Aktivsein prüfen
.seekActiveTime()	aktives Element animieren ab einem Zeitpunkt auf der Timeline wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	alle Media-Typen für Element zulässig
	siehe Eigenschaft .canSeek
.seekSegmentTime()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline ohne Wiederholung der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekTo()	aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline einschliesslich möglicher Wiederholungen der Animation wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.seekToFrame()	Frame eines aktiven Elementes auf der Timeline anwählen wenn Element nicht aktiv, so Fehler erzeugt per Eigenschaft .isActive das Element auf Aktivsein prüfen
	nicht alle Media-Typen für Element zulässig
	wenn unzulässig, so kein Fehler
	siehe Eigenschaft .canSeek
.segmentTimeToActiveTime()	Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
.segmentTimeToSimpleTime()	Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
.simpleTimeToSegmentTime()	Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren

Events:

onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element siehe onend und onrepeat
onend	erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount bzw. wenn das aktive Element gestoppt wird erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat und somit das Kindelement ebenfalls endet nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht siehe Methode .endElement() und Eigenschaft .end siehe onbegin und onrepeat
onmediacomplete	erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline siehe onmediaerror
onmediaerror	erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc. sinnvoll für Start des Elementes auf der Timeline ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet



	werden darf !
onoutofsync	siehe onmediacomplete erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes) sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked": Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()
onpause	siehe onsyncstored erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren auch bei body Objekt
onrepeat	erzeugt mit Start jeder Wiederholung der Animation des aktiven Elementes nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des ersten Durchlaufes, der keine Wiederholung ist nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen Handler für onrepeat kodiert hat (kein Heraufreichen des Ereignisses onrepeat zum Elternelement) .repeatCount bzw. .repeat muss > 1 sein: onrepeat wird also .repeatCount - 1 mal erzeugt Kind eines Elementes
onreset	siehe onend, onbegin, .repeatCount und .repeat erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(), also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht und damit zurückgesetzt wurde also nicht beim Initialisieren des Elementes und seiner Timeline durch Instanziierung des Elementes
onresume	erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird auch für body Objekt
onreverse	erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist) .autoReverse muss auf "true" stehen
onseek	erzeugt wenn eine Seek-Methode zum Element aktiviert wurde: .seekActiveTime() .seekSegmentTime() .seekTo() .seekToFrame()
onsyncstored	erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird nach vorausgegangenem Abbruch der Synchronisation siehe onoutofsync
ontrackchange	sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked" erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx) in der Playliste gewechselt wurde

4.3.2.2.4.3.39.24. Userdaten verwalten (.style.userData Behavior des Internet Explorer)

Userdaten per Datencache (UserData Store) dauerhaft oder befristet verwalten (**Achtung: Missbrauch möglich !**)

ab IE 5.x

Alternative : siehe Collection document.cookie

Umfang des Datenbereiches

Security Zone	Document Limit (KB)	Domain Limit (KB)
Local Machine	128	1024
Intranet	512	10240
Trusted Sites	128	1024
Internet	128	1024
Restricted	64	640

pro Objekt 1 Cache

Achtung: Cache überlebt das Schliessen einer Browserinstanz
kann durch andere Browserinstanz benutzt werden

alle CACHEDATEN werden in einem gemeinsamen Verzeichnis gespeichert
mit gemeinsamen Protokoll

Cache nicht manipulierbar möglich für Objekte HTML
HEAD
TITLE
STYLE

Cache muss im selben Ordner liegen wie Webpage

Bsp.: Wenn Cache im Ordner /private/
und Web page im Ordner /public/
so kann Webpage nicht auf den Cache zugreifen.

Verfallszeitpunkt als Zeitstempel für gecachte Daten:
per Eigenschaft .expires lesbar und setzbar



Daten werden **automatisch** durch den Browser gelöscht, wenn das Datum der Daten anhand des Zeitstempels als verfallen erkannt wurde, also der aktuelle Zeitpunkt laut PC-Uhr jünger ist, als der Zeitstempel.

Umwandlung in UTC-Zeitformat per Methode .toUTCString()

Syntax:

```
XML    <Prefix: CustomTag ID=kette STYLE="behavior:url('#default#userData')" >
HTML   <ELEMENT STYLE="behavior:url('#default#userData')" ID= kette >
Scripting object.style.behavior = "url('#default#userData')"
        object.addBehavior ("#default#userData")
```

Hinweis zur XML-Kodierung:

Anstelle der Tagbegrenzung>

sollte ein eigenständiges Endetag kodiert werden, also

.....>

....

</.....>

Prefix	laut XML-Namensraum
CustomTag	User-definierter Tag
kette	String mit dem ID muss kodiert werden

Beispiel:

```
<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachennamen definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache save
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);
```



```

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Eigenschaften:

.expires Zeitstempel der Daten im UserDataStore (User-Cache) per .style.userData Behavior
 Zeitstempel als Verfallszeitpunkt für automatische Löschung gecachter Daten:
 Daten werden **automatisch** durch den Browser gelöscht, wenn das Datum der Daten anhand des
 Zeitstempels als verfallen erkannt wurde, also der aktuelle Zeitpunkt laut PC-Uhr jünger ist, als
 der Zeitstempel.
 String im UTC (Universal Time Coordinate)-Format

Methoden:

.load() gespeicherten UserDataStore (User-Cache) auslesen per .style.userData Behavior
 Cache muss per Methode .save() zum Behavior gespeichert worden sein
.save() UserDataStore (User-Cache) speichern per .style.userData Behavior

4.3.2.2.4.3.39.25. HTML-Dokument mit Style-Sheet (CSS) im IE und NS

Nachfolgende Beschreibung ist die **HTML-konforme** Erweiterung des Objekt-Eigenschaft .style.

Aufgrund der HTML-Integration von CSS sollte der Browser CSS unterstützen. Ob diese Unterstützung vollständig ist, hängt vom Willen des Browserherstellers und seiner am Markt angebotenen Browserversionen ab. Dabei gilt: Auch wenn der HTML-Standard Style-Sheets implementiert hat und ab HTML 4.01 diverse Tags und Attribute als deprecated (missbilligt) setzt, weil sie vereinheitlicht durch genormte Stylesheets zu ersetzen sind, kann der Browser CSS nur im Umfang der implementierten Objekt-Eigenschaften von .style unterstützen. Wie immer gilt: Probieren geht über Studieren.

Microsoft setzt verstärkt auf XML, dessen Komponenten auf Basis der browsereigenen Objektimplementation mehr oder weniger HTML und CSS unterstützen. XML ist die perfektere Abbildung der browser-internen Objektstruktur und somit eine intern komplett objektorientierte Scriptsprache, allerdings hersteller- und browserspezifisch. Letztendlich wird CSS sinnigerweise als grundlegende Komponente und Eigenschaft **aller** Objekte integriert. Ein Vorteil, den Script-Programmierer beim Internet Explorer zu schätzen (lernen) wissen und den andere Browserhersteller mehr und mehr ihren Usern anbieten. CSS und .style bieten elementare Vielfalt bei der dynamischen Scriptprogrammierung, sind gewöhnungsbedürftig aber durch ihre Normung universell anwendbar.

Es sei darauf hingewiesen, dass die Verwendung von CSS-Klassen, Kodierung von <STYLE> bzw. dem STYLE-Attribut bzw. der Eigenschaften.style alle synonym sind. Vier Formen für dieselbe Sache, aber mit folgenden Unterschieden: CSS-Klassen können dokumentweit verwendet werden; <STYLE>, STYLE-Attribut und .style sind HTML-komponentenbezogen.

Hinweis: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

Bsp: zu .style: Es sollen DIVs erzeugt werden und deren Objektreferenzen (Zeiger) in einem
 Zeigerfeld gesammelt werden. Das Zeigerfeld dient der vereinfachten
 Verwaltung der dynamisch erzeugten DIV's.

```

var DIV_ID="Test_DIV"; // auch "Otto_DIV" etc.möglich , je nach Wunsch

var DIV_Zeiger=Array(); // sammelt alle ID als Zeichenketten

// DIV's dynamisch erzeugen und Zeiger einsammeln
var Left= 20; // Abstand vom linken Fensterrand in Pixeln
for (var i=0; i < 3; i++)
{
    Left+=10; // ab 30 Pixel vom linken Fensterrand
    document.write(
        '<DIV ID="' + DIV_ID + i.toString() + "' '
        + ' STYLE="position:absolute;'
        + 'left=' + Left + 'px;'
        + 'top=20px;'
        + '""
        + '>'
        + '</DIV>'
    );
    eval ('DIV_Zeiger[' + i + '] =' + DIV_ID + i.toString() );
}

```




```
// DIV's dynamisch auf dem Bildschirm bezüglich dem linken Fensterrand um 20 Pixel
// verschieben
Left= 40;
for (i=0; i < 3; i++)
{
    Left+=10; // ab 50 Pixel vom linken Fensterrand verschieben
    eval('document.all.DIV_Zeiger[' + i + '].style.left=' + Left);
}
```

Hinweis: Anstelle von i.toString() kann auch nur i kodiert werden, da der Browser aufgrund von document.write() i automatisch nach String umwandelt. analog für i.toString() innerhalb eval()

Die Verschiebung erfolgt natürlich mit den Objekten, welche innerhalb von <DIV> .. </DIV> kodiert wurden. Im Falle von IMG's innerhalb des DIV werden diese mit verschoben.

Vor allem **wegen** der Manipulation von HTML-Objekten werden DIV's verwendet.

4.3.2.2.4.3.39.25.1. Style-Sheet-Deklarations-Varianten

Werden Eigenschaften innerhalb von Tags kodiert, so gilt:

Sind Tags verschachtelt, so werden die Eigenschaften vererbt

--> Überschreiben innerhalb der Erb-Ebene möglich

Kommentierung nur innerhalb 1 Zeile per

nicht jeder Browser unterstützt Style-Sheet, also Style-Sheets innerhalb von <!-- --> kodieren

für Tags wird nicht unterschieden zwischen Groß und Klein

4.3.2.2.4.3.39.25.1.1. Style-Sheet Eigenschaften innerhalb <HEAD> deklarieren

```
<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--
            @eigenschaften                                /* festkodiert ist @
        -->
    </STYLE>
</HEAD>
<BODY>
....
</BODY>
```

Bsp: @import

4.3.2.2.4.3.39.25.1.2. Style-Sheet-Format und Style-Sheet-Eigenschaften als tag-unabhängig deklarieren (Attribut ID)

```
<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--
            #format_name{eigenschaften_liste} /* festkodiert ist #
        -->
    </STYLE>
</HEAD>
<BODY>
    <tag_name ID="format_name"> ..... </tag_name>
</BODY>
```

format_name: Bsp: fettkursiv

innerhalb STYLE mit vorgesetztem # kodieren !

innerhalb BODY ohne # kodieren !

eigenschaften_liste: Semikolontrennung

muss mit Semikolon enden

wenn Blank enthalten, so Liste in " " setzen

Bsp: {font-weight:bold;font-style:italic;}

Hinweise: ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das

tag-unabhängige ersetzt.

tag-unabhängige Style-Sheet-Format fügen sich ansonsten zu den tag-abhängigen hinzu.

4.3.2.2.4.3.39.25.1.3. Style-Sheet-Schlüsselwort und Style-Sheet-Eigenschaften als tag-abhängig deklarieren

```
<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--
            tag_name:schlüsselwort{eigenschaften_liste} /* Doppelpunkt ist festkodiert
        -->
```



```

</STYLE>
</HEAD>
<BODY>
    <tag_name> ..... </tag_name>
</BODY>

```

tag_name: Schlüsselwort
Bsp: a:link

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font-style:italic;}

Hinweise: ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das tag-unabhängige ersetzt.
tab-unabhängige Style-Sheet-Format fügen sich ansonsten zu den tag-abhängigen hinzu.

4.3.2.2.4.3.39.25.1.4. *Style-Sheet Eigenschaften ohne Tag-Attribut ID deklarieren*

```

<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--
            tag_namen_liste{eigenschaften_liste}

        //-->
    </STYLE>
</HEAD>
<BODY>
</BODY>

```

tag_namen_liste: Kommatrennung
alle Tags haben genau eine gemeinsame Eigenschaftsliste
Bsp: h1,h2

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font-style:italic;}

4.3.2.2.4.3.39.25.1.5. *Style-Sheet-Unterklasse deklarieren*

```

<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--
            tag_name.unterklasse_name{eigenschaften_liste}    /* Punkt ist festkodiert
            oder all.unterklasse_name{eigenschaften_liste}     /* Punkt und all sind festkodiert
            oder .unterklasse_name{eigenschaften_liste}        /* Punkt ist festkodiert

        //-->
    </STYLE>
</HEAD>
<BODY>
    <tag_name CLASS="unterklasse_name"> ..... </tag_name>
</BODY>

```

tag_namen.unterklasse_name: Bsp: p.normal
all.unterklasse_name: Unterklasse gilt für ALLE Tags
.unterklasse_name: identisch mit all.unterklasse_name

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font-style:italic;}

4.3.2.2.4.3.39.25.1.6. *Style-Sheet-Eigenschaften mit Attribut STYLE deklarieren*

```

<HEAD>
    <TITLE>.... </TITLE>
    <STYLE TYPE="text/css">
        <!--

        //-->
    </STYLE>
</HEAD>
<BODY>
    <tag_name STYLE=eigenschaften_liste>..... </tag_name>
</BODY>

```



Bsp: <P STYLE= > </P>

eigenschaften_liste: Semikolontrennung

muss mit Semikolon enden

wenn Blank enthalten, so Liste in " " setzen

Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.1.7. *Style-Sheet-Eigenschaften eines HTML-Elementes deklarieren*

```
<HEAD>
  <TITLE>.... </TITLE>
  <STYLE TYPE="text/css">
    <!--

      //-->
    </STYLE>
</HEAD>
<BODY>
  <tag_name ..>
    <SPAN STYLE=eigenschaften_liste>
      zu formatierendes HTML-Element
    </SPAN>

  ....
  </tag_name>
</BODY>
```

Bsp: <P STYLE= > </P>

eigenschaften_liste: Semikolontrennung

muss mit Semikolon enden

wenn Blank enthalten, so Liste in " " setzen

Bsp: {font-weight:bold;font_style:italic;}

HTML-Element MUSS Endetag besitzen, das kodiert werden MUSS

Bsp: <P STYLE="font-size:12pt;"> </P>

4.3.2.2.4.3.39.25.1.8. *Beispiele für Style-Sheet*

Beispiel 1

```
<HEAD>
  <TITLE> ..... </TITLE>
  <STYLE TYPE="text/css">
    <!--

      body{    background-color:#FFFFFF;
               background-image:url(test.jpg);
               background-repeat:repeat-y;
               margin-top: 1cm;
               margin-left: 2cm;
               margin-right: 1cm;

               }
      p{       font-family:serif;
               font-size: 12pt;
               text-align:justify;

               }
      p.zusatz_format_p_tag{text-indent:0.5cm;}
      .zusatz_format_alle_tags{
               position:absolute;
               top:4cm;
               z-index:3;
               font-size:40pt;
               font-width:bold;
               color:blue;
               text-align:center;}

    -->
  </STYLE>
</HEAD>

<BODY>
  <SPAN CLASS="zusatz_format_alle_tags"> text </SPAN>
  <P CLASS="zusatz_format_p_tag" text </P>
</BODY>
```

<!-- --> für Browser kodieren, die CSS nicht können

background-image: Hintergrundbild test.jpg verwenden

Achtung: Möglichst absolute Urls bzw. Pfadangaben verwenden, da
eventuell Browser relative Angaben nicht interpretieren



		kann
	Bsp.:	relativ :url('ordner/test.gif')
		absolut :url('www.test.de/ordner/test.gif')
background-repeat:	repeat-y	Hintergrundbild senkrecht wiederholen
	repeat-x	Hintergrundbild waagerecht wiederholen
	no-repeat	Hintergrundbild nicht wiederholen
p{....}		gilt für JEDES P-Tag ohne CLASS-Attribut zu kodieren
margin-xxxx		Seitenrand
text-align		Textausrichtung
text-indent		Einzug links
position		Art der position-Angabe folgenden Angaben
	Bsp:	position:absolute --> Absolutangaben
		top:4cm hier 4 cm
top		Abstand zum Browserfenster oben
z-index		Angezeigte Schicht (analog zu LAYER)
		--> ab 1 = unterste Schicht
		verwenden für überlappende Textbereiche
p.zusatz_format_p_tag{ .. }		per CLASS-Attribut anwenden im P-Tag

Beispiel 2

```

<HTML>
<HEAD>
<TITLE> ..... </TITLE>
<STYLE TYPE="text/css">
<!--
      .eigene_css_klasse{ ..... }
-->
</STYLE>
</HEAD>
<BODY>
<DIV CLASS=eigene_css_klasse>.....</DIV>
.....
</BODY>
</HTML>

```

4.3.2.2.4.3.39.25.2. Style-Sheet und numerische (nicht vordefinierte) Farbangaben

#rrgbb oder rgb(rrr,ggg,bbb) oder vordefinierte Farben

Rot-Anteil:

hexa	rr
dezimal 0-255	rrr

Grün-Anteil:

hexa	gg
dezimal 0-255	ggg

Blau-Anteil:

hexa	bb
dezimal 0-255	bbb

4.3.2.2.4.3.39.25.3. Style-Sheet und vordefinierte Bezeichner

Die Nachfolgende Beschreibung umfasst nur die wichtigsten Style-Sheet-Eigenschaften.

4.3.2.2.4.3.39.25.3.1. Style-Sheet und vordefinierte Dimensionen

pt	Point	=	1/72 inches
pc	Pica	=	12 pt
in	Inch	=	2,54 cm
mm			
cm			
px	Pixel		
em	relativ zur Schrifthöhe des Elementes, das per CSS formatiert wird		
ex	relativ zur Höhe des Buchstaben Grosses X		
%	Prozentanteil von der Norm des per CSS formatierten Elementes		

4.3.2.2.4.3.39.25.3.2. Style-Sheet- und Dezimalkomma

Dezimalkomma ist der Punkt und NICHT das Komma !!!

4.3.2.2.4.3.39.25.3.3. Style-Sheet und vordefinierte Schriften (Font und Style)

vordefinierte Schriftarten:	serif
	san-serif
	cursive
	fantasy
	monospace

vordefinierte Style:	italic	kursiv
	oblique	kursiv
	normal	nicht kursiv



4.3.2.2.4.3.39.25.3.4.	<i>Style-Sheet und vordefinierte Farbbereiche</i>	
4.3.2.2.4.3.39.25.3.4.1.	<i>Style-Sheet und Farbe des Desktop</i>	
background	Farbe	Hintergrund Desktop
4.3.2.2.4.3.39.25.3.4.2.	<i>Style-Sheet und Farbe des Dokumentfensters</i>	
window	Farbe	Hintergrund Dokumentfenster
windowframe	Farbe	Rahmen Dokumentfenster
windowtext	Farbe	Text im Dokumentfenster
4.3.2.2.4.3.39.25.3.4.3.	<i>Style-Sheet und Farbe aktives Fenster</i>	
activeborder	Farbe	Farbe aktiven Fenstertitelzeile (-leiste) oben
activecaption	Farbe	Farbe Überschrift in der aktiven Fenstertitelzeile
appworkspace	Farbe	Farbe Hintergrund aktives Fenster
4.3.2.2.4.3.39.25.3.4.4.	<i>Style-Sheet und Farbe inaktives Fenster</i>	
inactiveborder	Farbe	Farbe nichtaktive Fenstertitelzeile
inactivecaption	Farbe	Farbe Überschrift nichtaktive Fenstertitelzeile
4.3.2.2.4.3.39.25.3.4.5.	<i>Style-Sheet und Farbe des Dialogfensters</i>	
captiontext	Farbe	Farbe Überschrift im Dialogfenster
greytext	Farbe	Farbe deaktivierter Text im Dialogfenster
buttonface	Farbe	Farbe Button im Dialogfenster
buttonhighlight	Farbe	Farbe 3D-Schatten von Button im Dialogfenster
buttontext	Farbe	Farbe Button-Text im Dialogfenster
4.3.2.2.4.3.39.25.3.4.6.	<i>Style-Sheet und Farbe einer Auswahlliste</i>	
highlight	Farbe	Farbe ausgewählter Eintrag in Auswahlliste
highlighttext	Farbe	Farbe selektierter Text in Auswahlliste
4.3.2.2.4.3.39.25.3.4.7.	<i>Style-Sheet und Farbe von Tooltip und Popuphilfe (Hint)</i>	
infobackground	Farbe	Farbe Tooltips und Popuphilfen (Hints)
infotext	Farbe	Farbe Text von Tooltips und Popuphilfen (Hints)
4.3.2.2.4.3.39.25.3.4.8.	<i>Style-Sheet und Farbe eines Menü</i>	
menu	Farbe	Farbe Menüleiste
menutext	Farbe	Farbe Menüeintrag
4.3.2.2.4.3.39.25.3.4.9.	<i>Style-Sheet und Farbe der Scrolleiste</i>	
scrollbar	Farbe	Farbe Scrolleiste
4.3.2.2.4.3.39.25.3.4.10.	<i>Style-Sheet und Farbe eines 3D-Elements</i>	
threeddarkshadow	Farbanteil dunkel	für Schatten bei 3D-Element
threedface	Farbe	Farbe 3D-Element
threedhighlight	Farbe	Farbe gerade angeklicktes 3D-Element
threedlightshadow	Farbanteil hell	für Schatten bei 3D-Element
threedshadow	Farbanteil dunkel	für Schatten bei 3D-Element
4.3.2.2.4.3.39.25.4.	<i>Style-Sheet-Dateien</i>	
4.3.2.2.4.3.39.25.4.1.	<i>Style-Sheet-Schriftdatei laden (*.eot bzw. *.pfr)</i>	
Microsoft:	*.eot	
Netscape	*.pfr	

```

<STYLE TYPE="text/css">
  <!--
    @font-face{font-family:familien_name;
                src: url(eot_bzw._prf_datei_liste);}
    font-size: ...
    ...
  // -->
</STYLE>

```

4.3.2.2.4.3.39.25.4.2. *Style-Sheet-Informationen aus Datei einbinden (*.css)*

Variante 1:

```

<LINK
  REL=stylesheet
  TYPE="text/css"
  MEDIA="typ_name"
  HREF="url_oder_css_dateiname"
>
<STYLE TYPE="text/css">
  <!-- weitere Stylesheet-Angaben
  // -->
</STYLE>

```

typ_name: Name des Ausgabe-Mediums
 "all" alle Medien
 "screen" Bildschirm



"print" Drucker

Variante 2:

```
<LINK
    REL=stylesheet
    TYPE="text/css"
    [MEDIA="typ_name"]          optional
    HREF="url_oder_css_dateiname"
>

<STYLE TYPE="text/css">
    <!--
        @import (url_oder_css_datei) typ_liste;
        ....
    // -->
</STYLE>

typ_liste: Liste der Ausgabe-Medien
           Kommattrennung
           Bsp: print,screen;
```

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x und NS 6.x):

Seitenteile, die nicht gedruckt werden sollen, in <DIV CLASS="keindruck"> und </DIV> oder in und einschliessen.

zwischen <HEAD> und </HEAD> einfügen:

```
<LINK REL="stylesheet" MEDIA="print" HREF="print.css">
```

print.css enthält nur
.keindruck {display:none;}

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

Seitenteile, die nur auf Ausdrucken sichtbar sein sollen, in <DIV CLASS="nurdruck"> und </DIV> oder in und einschliessen

zwischen <HEAD> und </HEAD> einfügen:

```
<LINK REL="stylesheet" MEDIA="screen" HREF="screen.css">
```

screen.css enthält nur
.nurdruck {display:none;}

4.3.2.2.4.3.39.25.4.3. *Style-Sheet und Ausgabemedien anhand browserinterner Pseudoklassen*

4.3.2.2.4.3.39.25.4.3.1. *Ausgabemedien-spezifische CSS-Datei importieren (@import)*

@import (css_datei) typ_liste;

typ_liste: Liste der Ausgabe-Medien
 Kommattrennung
 Bsp: print, screen;

4.3.2.2.4.3.39.25.4.3.2. *Ausgabenmedien-spezifische Style-Sheet-Eigenschaften deklarieren (@media)*

@media typ_liste{eigenschaften_liste}

typ_liste: Liste der Ausgabe-Medien
 Kommattrennung
 Bsp: print, screen;
eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " setzen
 Bsp: {font-weight:bold;font-style:italic;}

4.3.2.2.4.3.39.25.4.4. *Style-Sheet und Font-Dateien (@font-face) anhand browserinterner Pseudoklasse*

@font-face {eigenschaften_liste;

 src: url (url_liste_mit_kommattrennung);
 und/oder local (datei_namen_liste_mit_blank_trennung);
 und/oder format (TrueType);
 }

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " setzen
 Bsp: {font-weight:bold;font-style:italic;}

4.3.2.2.4.3.39.25.4.5. *Style-Sheet und Seiten-Eigenschaften (@page) anhand browserinterner Pseudoklasse*



@page:first {eigenschaften_liste} Regel gehört der 1. Seite
 @page:footer{eigenschaften_liste} Regel gehört der Fußnote
 @page:header{eigenschaften_liste} Regel gehört der Kopfnote
 @page:left{eigenschaften_liste} Regel gehört der linken Seite
 @page:left : header{eigenschaften_liste} Regel gehört der Kopfnote Seite links
 @page:left : footer{eigenschaften_liste} Regel gehört der Fußnote Seite links
 @page:right{eigenschaften_liste} Regel gehört der rechten Seite
 @page:right:header{eigenschaften_liste} Regel gehört der Kopfnote Seite rechts
 @page:right:footer{eigenschaften_liste} Regel gehört der Fußnote Seite rechts

eigenschaften_liste: Semikolontrennung
 muss mit Semikolon enden
 wenn Blank enthalten, so Liste in " " setzen

Beispiel: @page : left {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.4.6. *Style-Sheet und Hintergrund-Grafik*

background-image: url(grafik_datei); Hintergrunddatei GIF oder JPG
 background-color:#rrgbbb; oder rgb(rrr,ggg,bbb); Hintergrundfarbe
 background-repeat: repeat; Hintergrundgrafik kacheln auf gesamten Hintergrund
 oder repeat-x; Hintergrundgrafik für 1 Zeile kacheln
 oder repeat-y; Hintergrundgrafik für 1 Spalte kacheln
 oder no-repeat; kein Kacheln des Hintergrundbildes
 background-attachment: scroll; Hintergrundgrafik scrollt mit Vordergrund
 oder fixed; Hintergrundgrafik scrollt nie
 background-position: top; Hintergrundgrafik auf Oberkante Hintergrund
 oder center; Hintergrundgrafik zentriert auf Hintergrund
 oder middle; Hintergrundgrafik mittig auf Hintergrund
 oder bottom; Hintergrundgrafik auf Unterkante Hintergrund
 oder left; Hintergrundgrafik linksbündig auf Hintergrund
 oder right; Hintergrundgrafik rechtsbündig auf Hintergrund
 background: eigenschaften_liste_mit_blank_trennung; obige Eigenschaften als Liste
 Bsp: background: url(back.gif) no-repeat middle; Blank-Trennung !!

-moz-opacity:faktor Transparenz des Hintergrundbildes
 nur NS 6.x
 Faktor > 0 und bis 1
 Bsp: 0.9 entspricht 90%
 Bsp:

4.3.2.2.4.3.39.25.4.7. *Style-Sheet und Rahmen-Eigenschaften (Border)*

border-top-width: rahmen_dicke_von_rahmen_oberhalb_des_elementes;
 border-bottom-width: rahmen_dicke_von_rahmen_unterhalb_des_elementes;
 border-left-width: rahmen_dicke_von_rahmen_links_des_elementes;
 border-right-width: rahmen_dicke_von_rahmen_rechts_des_elementes;

border-color:#rrgbbb; oder rgb(rrr,ggg,bbb); Rahmenfarbe

border-style: none; kein Rahmen um das Element
 oder dotted; gepunkteter Rahmen um das Element
 oder dashed; gestrichelter Rahmen um das Element
 oder solid; einfacher durchgezogener Rahmen um das Element
 oder double; doppelter durchgezogener Rahmen um das Element
 oder groove; 3D-Effekt 1
 oder ridge; 3D-Effekt 2
 oder inset; 3D-Effekt 3
 oder outset; 3D-Effekt 4

border-top: eigenschaften_liste_1;
 border-bottom: eigenschaften_liste_2;
 border-right: eigenschaften_liste_3;
 border-left: eigenschaften_liste_4;
 border-width: eigenschaften_liste_5;

border:eigenschaften_liste_6;

Hinweise:

rahmen_dicke_.... : auch thin oder medium oder thick

eigenschaften_liste_1: wert_von_border-top-width #rrgbbb wert_von_border-style
Werte mit Blank trennen

eigenschaften_liste_2: wert_von_border-bottom-width #rrgbbb wert_von_border-style
Werte mit Blank trennen



eigenschaften_liste_3:	wert_von_border-left-width #rrgbbb wert_von_border-style Werte mit Blank trennen
eigenschaften_liste_4:	wert_von_border-right-width #rrgbbb wert_von_border-style Werte mit Blank trennen
eigenschaften_liste_5:	Werte-Liste aller möglichen Bordereigenschaften Werte mit Blank trennen

Anstelle von Doppelpunkt ist auch das Gleichheitszeichen kodierbar

Bsp: border-top: thick inset rgb(192,192,255);

4.3.2.2.4.3.39.25.4.8. *Style-Sheet und HTML-Tag-bezogene Eigenschaften*

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font-style:italic;}

4.3.2.2.4.3.39.25.4.8.1. *Style-Sheet-Eigenschaften zu <A>*

a:link{eigenschaften_liste}	Eigenschaften für noch nicht besuchte Links
a:visited{eigenschaften_liste}	Eigenschaften für schon besuchte Links
a:active{eigenschaften_liste}	Eigenschaften für gerade besuchtes Link
a:hover{eigenschaften_liste}	Eigenschaften für Maus über Link , ab IE 4.x nur für Tags mit HREF-Attribut

4.3.2.2.4.3.39.25.4.8.2. *Style-Sheet-Eigenschaften zu <P>*

p:first-line{eigenschaften_liste}	Eigenschaften Absatz 1. Zeile
p:first-letter{eigenschaften_liste}	Eigenschaften Absatz 1. Zeile, 1. Zeichen
p:before{content:"freier_text" }	Text vor dem Element einfügen
p:after{content:"freier_text" }	Text nach dem Element einfügen

4.3.2.2.4.3.39.25.4.8.3. *Style-Sheet-Eigenschaften zu <H1> bis <H6>*

h1:first-line{eigenschaften_liste}	Eigenschaften Überschrift 1. Zeile
h1:first-letter{eigenschaften_liste}	Eigenschaften Überschrift 1. Zeile, 1. Zeichen

für H2 bis H6 analog

Beim Internet Explorer 6.0 unter Windows 98 bzw. Windows XP wurde die Darstellung der Überschriften verändert, wenn der jeweilige Standardfont benutzt wird. Empfehlung: Neudefinition der <Hx>-Tags per Style (soweit möglich).

4.3.2.2.4.3.39.25.5. *Style-Sheet-Eigenschaften (Style-Sheet-Formatangaben)*

4.3.2.2.4.3.39.25.5.1. *Style-Sheet und Wertzuweisung an Eigenschaften*

eigenschaft: wert; oder eigenschaft=wert;

Hinweis: Pflichtkodierung von Doppelpunkt bzw. Gleichheitszeichen
Semikolon

Folge von Eigenschaften möglich: als Liste mit Semikolontrennung

Bsp: eigenschaft1=wert1;; eigenschaft_n=wert_n;

wenn Blank in Kodierung enthalten, so alles in " " setzen

Bsp: style="font-style: italic; color:red;"

4.3.2.2.4.3.39.25.5.2. *Style-Sheet-Eigenschaften für HTML-Elemente*

4.3.2.2.4.3.39.25.5.2.1. *Style-Sheet und Auswertung von Pixelpositions-Angaben*

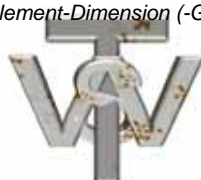
position: absolute;	Positionsangaben als bezüglich Anzeigefenster-Rand Element ist scrollbar
position: fixed;	Positionsangaben bezüglich Anzeigefenster-Rand Element ist nicht scrollbar
position: relative;	Positionsangaben als bezüglich Vorgänger-Element
position: static;	hebt absolute bzw. fixed bzw. relative auf

4.3.2.2.4.3.39.25.5.2.2. *Style-Sheet und Abstand von HTML-Elementen*

top:abstand_obenhalb_in_pixel;	oder	top:auto;
left:abstand_links_in_pixel;	oder	left:auto;
bottom:abstand_unterhalb_in_pixel;	oder	bottom:auto;
right:abstand_rechts_in_pixel;	oder	right:auto;

padding-top: abstand_element_grenze_oben_zum_element_obenhalb_in_pixel;
padding-bottom: abstand_element_grenze_unten_zum_element_unterhalb_in_pixel;
padding-left: abstand_element_grenze_links_zum_element_links_in_pixel;
padding-right: abstand_element_grenze_rechts_zum_element_rechts_in_pixel;
padding: abstand; gilt für links, rechts, oben und unten zugleich

4.3.2.2.4.3.39.25.5.2.3. *Style-Sheet und HTML-Element-Dimension (-Grenzen, -Anzeigebereich)*



width: breite_in_pixel;	oder	width:auto;
min-width: minimale_breite_in_pixel;	oder	min-width:auto;
max-width: maximale_breite_in_pixel;	oder	max-width:auto;

height: hoehe_in_pixel;	oder	height:auto;
min-height: minimale_hoehe_in_pixel;	oder	min-height:auto;
max-height: maximale_hoehe_in_pixel;	oder	max-height:auto;

overflow:hidden;	wenn Elementgröße > max-width und max-height, so wird Element auf Maximalwerte begrenzt
overflow:visible;	wenn Elementgröße > max-width und max-height , so werden Maximalwerte ignoriert

4.3.2.2.4.3.39.25.5.2.4. Style-Sheet und HTML-Element-Sichtbarkeit

visibility:hidden;	unsichtbar, aber Platzhalter möglich
visibility:visible;	sichtbar

display:	none;	unsichtbar	UND kein Platzhalter möglich
		block;	sichtbar
		inline;	sichtbar

4.3.2.2.4.3.39.25.5.2.5. Style-Sheet und Element-Ausschnitt

```
clip: rect (oben rechts unten links); Pixelangaben bezüglich Elementgrenze
clip: auto;
```

4.3.2.2.4.3.39.25.5.2.6. Style-Sheet und Element-Scrolling

Scrolling nötig, wenn Elementgröße > max-width und max-height
ABER Maximalwerte eingehalten werden sollen

overflow:scroll;	Scrolling immer möglich
overflow:auto;	

siehe auch overflow: visible; bzw. overflow hidden;

4.3.2.2.4.3.39.25.5.2.7. *Style-Sheet und Layer-Element*

z-index:index_nr;	Nummer innerhalb der Anzeigereihenfolge sich überlappender Elemente ab 1 = unterste Schicht
-------------------	--

4.3.2.2.4.3.39.25.5.2.8. Style-Sheet und HTML-Elemente-Anordnung im Dokument

direction:ltr;	von links nach rechts; ist Standard
direction: rtl;	von rechts nach links

<code>display:block;</code>	Element in eigenem Absatz anzeigen
<code>display:inline;</code>	Element nicht in eigenem Absatz anzeigen
<code>display:none;</code>	Element nicht anzeigen
<code>display:list-item;</code>	

float:left;	Element linksbündig; Textfluss rechts
float:right;	Element rechtsbündig; Textfluss links
float:none;	Standard

clear:left;	hebt float: left; auf
clear:right;	hebt float: right; auf
clear:both;	hebt float: left; UND float: right; auf
clear:none;	identisch mit float: none;

4.3.2.2.4.3.39.25.5.2.9. *Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet)*

display: list-item; Element in eigenem Absatz UND mit Bullet anzeigen

4.3.2.2.4.3.39.25.5.3. *Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung)*

list-style-type:	decimal;	1	2	3 ...
oder	lower-roman;	i	ii	iii ..
oder	lower-alpha	a	b	c
oder	upper-roman;	I	II	III ...
oder	upper-alpha;	A	B	C
oder	disk;	Bullet ist Dateisymbol		
oder	circle;	Bullet ist rund		
oder	square;	Bullet ist rechteckig		
oder	none;			

list-style-position:inside;	Listenelement einrücken; ist Standard
list-style-position:outside	Listenelement ausrücken

list-style-image: url(bullet_grafik_datei); GIF oder JPG

list-style: liste aller obigen eigenschaften mit blank trennung;

4.3.2.2.4.3.39.25.5.4. *Style-Sheet-Eigenschaften für Tabelle*



caption-side:	top;	Überschrift oberhalb zentriert
oder	topleft;	Überschrift oberhalb linksbündig
oder	topright;	Überschrift oberhalb rechtsbündig
oder	bottom;	Überschrift unterhalb zentriert
oder	bottomleft;	Überschrift unterhalb linksbündig
oder	bottomright;	Überschrift unterhalb rechtsbündig

row-span: anzahl_der_zeilen_über_die_sich_die_zelle_ausstrecken_soll;

column-span: anzahl_der_spalten_über_die_sich_die_zelle_ausstrecken_soll;

4.3.2.2.4.3.39.25.5.5. *Style-Sheet-Eigenschaften für Seitendarstellung im Dokument*

margin-top:	rand_breite_oben;	z.B. 2cm
margin-bottom:	rand_breite_unten;	z.B. 2cm
margin-left:	rand_breite_links;	z.B. 2cm
margin-right:	rand_breite_rechts;	z.B. 2cm

margin: werte_liste_obiger_breiten_mit_blank_trennung;

wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten 2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für links UND rechts 3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben 2. Wert gilt für rechts 3. Wert gilt für unten 4. Wert gilt für links

size:	seiten_breite;	z.B. 20cm
oder	seiten_hoehe;	z.B. 29cm
oder	seiten_breite seiten_hoehe;	Blanktrennung
oder	landscape;	Querformat
oder	portrait;	Hochformat
oder	auto;	

page-break_before:	always;	immer Seitenumbruch vor aktuellem Element erzeugen
oder	aroid;	nie Seitenumbruch vor aktuellem Element erzeugen
oder	left;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann linksbündig ablegen
oder	right;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann rechtsbündig ablegen
oder	auto;	
page-break_after:	always;	immer Seitenumbruch nach aktuellem Element erzeugen
oder	aroid;	nie Seitenumbruch nach aktuellem Element erzeugen
oder	left;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element linksbündig ablegen
oder	right;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element rechtsbündig ablegen
oder	auto;	

4.3.2.2.4.3.39.25.5.6. *Style-Sheet-Eigenschaften für Text*

text-indent: wert;
wert > 0, so Textzeile einrücken
wert < 0, so Textzeile ausrücken

text-align:	left;	Text linksbündig
oder	center;	Text zentriert
oder	right;	Text rechtsbündig
oder	justify;	Text im Blocksatz
vertical-align:	top;	Text auf Oberkante der Umgebung
oder	middle;	Text auf Mitte der Umgebung
oder	bottom;	Text auf Unterkante der Umgebung
oder	sub;	Text tieferstellen
oder	super;	Text höher stellen
oder	baseline;	Text auf Basislinie des umgebenden Textes mit verschiedenen Schriftarten
oder	text-top;	Text auf obere Linie des umgebenden Textes mit verschiedenen Schriftarten
oder	text-bottom;	Text auf untere Linie des umgebenden Textes mit verschiedenen Schriftarten

font-size: schriftgroesse_wert; z.B. 2pt oder vordefiniert

line-height: zeilenhoehe_wert; z.B. 2pt* oder vordefiniert/



white-space: normal; automatischer Zeilenumbruch einschalten
 oder pre; manuellen Zeilenumbruch einschalten
 oder nowrap; kein Zeilenumbruch möglich

color:#rrggbb; oder rgb(rrr,ggg,bbb); auch vordefiniert möglich

columns: anzahl_der_text_spalten_fuer_textfluss;
 column-gap: abstand_der_text_spalten_fuer_textfluss;
 column-rule-width: dicke_des_trennstriches_zwischen_den_textspalten_vom_textfluss;
 column-rule-color: #rrggbb; oder rgb(rrr,ggg,bbb); Farbe Trennstrich
 column-rule-style: none; kein Trennstrich zwischen Textspalten
 oder dotted; gepunkteter Trennstrich
 oder dashed; gestrichelter Trennstrich
 oder solid; einfacher durchgezogener Trennstrich
 oder double; doppelter durchgezogener Trennstrich
 oder groove; 3D-Effekt 1
 oder ridge; 3D-Effekt 2
 oder inset; 3D-Effekt 3
 oder outset; 3D-Effekt 4

column-rule: werte_liste_aus_obigen_textfluss_werten_mit_blank_trennung;
 word-spacing: abstand_zwischen_woertern_im_text;
 letter-spacing: abstand_zwischen_zeichen_im_text;
 text-decoration: underline; Text unterstrichen
 oder overline; Text überstrichen
 oder line-through; Text durchgestrichen
 oder blink; Text blinkend
 oder none; Text normal

text-transform: capitalize; Wortanfang ALLER Wörter auf Großbuchstaben
 oder uppercase; alle Zeichen nach Großbuchstaben
 oder lowercase; alle Zeichen nach Kleinbuchstaben
 oder none;

text-shadow: #rrggbb; oder rgb(rrr,ggg,bbb); auch vordefiniert möglich
 oder none; kein Textschatten

orphans: anzahl_der_VOR_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
 Schusterjunge

widow: anzahl_der_NACH_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
 Hurenkind

4.3.2.2.4.3.39.25.5.7. Style-Sheet-Eigenschaften für Font

font-family: schriftarten_liste_mit_kommatrennung;

es wird der ERSTE auf dem lokalen Rechner gefundene Font aus der
 Liste geladen

wenn blank vorhanden, so alles in " " setzen

vordefinierte Schriftarten:

serif
 san-serif
 cursive
 fantasy
 monospace

font-style: italic; kursiv
 oder oblique; kursiv
 oder normal; nicht kursiv

font-variant: small-caps; kleine Großbuchstaben (Kapitälchen)
 oder normal; kein Kapitälchen

font-size: schrift_groesse_wert;
 oder xx-small; winzig
 oder x-small; sehr klein
 oder small; klein
 oder medium; mittel
 oder large; groß
 oder x-large; sehr-groß
 oder xx-large; riesig
 oder smaller; etwas kleiner als normal
 oder larger; etwas größer als normal

font-weight: bold; fett
 oder bolder; extra fett
 oder lighter; dünn
 oder normal; nicht dünn und nicht fett
 oder 100; extra dünn



oder	200;	
oder	300;	
oder	400;	
oder	500;	medium
oder	600;	
oder	700;	bold
oder	800;	
oder	900;	extra fett

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;

4.3.2.2.4.3.39.25.5.8. Style-Sheet-Eigenschaften für Mauscursor

cursor:	auto;	
oder	default;	
oder	crosshair;	Fadenkreuz
oder	pointer;	Zeiger
oder	move;	Kreuz für Beweglichkeit
oder	n-resize;	Pfeil Nord
oder	ne-resize;	Pfeil Nord-Ost
oder	e-resize;	Pfeil Ost
oder	se-resize;	Pfeil Süd-Ost
oder	s-resize;	Pfeil Süd
oder	sw-resize;	Pfeil Süd-West
oder	w-resize;	Pfeil West
oder	nw-resize;	Pfeil Nord-West
oder	text;	
oder	wait;	Sanduhr
oder	help;	Fragezeichen
oder	url(url_oder_mauscursor_grafik_datei);	GIF oder JPG

4.3.2.2.4.3.39.25.5.9. Style-Sheet-Eigenschaften für Unicode (Zeichensatz)

unicode-range:U+xxxx-yyy;

Fragezeichen als Joker innerhalb von xxxx und yyyy verwendbar

Bsp: unicode-range:U+0000-007F; ist ASCII-Zeichensatz
 unicode-range:U+0000-00?F; ? steht für 0 bis F --> ist nicht ASCII-Zeichensatz

4.3.2.2.4.3.39.25.6. Style-Sheet-Beispiele**Beispiel 1**

```
<STYLE TYPE="text/css">
<!--
h1 {font-size:24pt;margin-top:1.2cm;margin-left:30px;}
body {background-color:rgb(51,0,102);}
p,li {font-size:12pt;line-height:14pt;font-family:Helvetica,Arial;}
p.normal {font-size:10pt;color:black;}
p.klein {font-size:8pt;color:black;}
all.rot {color:red;}      oder .rot {color:red;}
                                Rot-Definition für ALLE Tags
                                Anwendung z.B. per <P CLASS="normal">text</P>
                                Anwendung z.B. per <P CLASS="rot">text</P>
                                Anwendung z.B. per <H1 CLASS="rot">text</H1>
#fett_kursiv {font-weight:bold;font-style:italic;}
                                Anwendung z.B. per <P ID="fett_kursiv">text</P>
a:link {color:#FF0000;font-weight:bold;}
                                Anwendung z.B. per <A HREF=".....">text</A>
// -->
</STYLE>
```

Anwendung je nach Tag --> meist innerhalb <BODY>

Beispiel 2

```
<BODY>
...
<DIV STYLE="background-color:#FF0000;"> ..... </DIV>
<P>
    text1
    <SPAN STYLE="color:red;">
        text2_in_rot;
    </SPAN>
    text3_wie_text1
</P>
```

....



</BODY>

4.3.2.2.4.3.39.26. Beispiele für Objekteigenschaft .style

Kodierung der Style-Sheets:

innerhalb CSS-Klasse: immer mit Bindestrich

als .style-Eigenschaft ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabe

Bsp: in CSS-Klasse background-color
 .style.backgroundColor

alle Eigenschaften sind les- und schreibbar

Die Erweiterung von .style durch HTML-CSS-konforme Style-Sheets ist möglich, aber zu prüfen, ob sie auch unterstützt werden. Beachte die o.g. Kodierungsregel.

4.3.2.2.4.3.39.26.1. Zeichensatz- und Texteigenschaften

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;
 Liste darf maximal 6 der nachfolgenden Eigenschaften beinhalten

caption	Zeichensatz für Objekte mit Überschriften
font-style	
font-variant	
font-weight	
font-size	
line-height	
font-family	
icon	Zeichensatz für Grafik
menu	Zeichensatz in Menüs
message-box	Zeichensatz in Messages-Boxen
small-caption	Zeichensatz für Control-Elemente
status-bar	Zeichensatz für Statuszeile

font-family: schriftarten_liste_mit_kommatrennung;
 es wird der ERSTE auf dem lokalen Rechner gefundene Font aus der
 Liste geladen
 wenn blank vorhanden, so alles in " " setzen
 vordefinierte Schriftarten: serif
 san-serif
 cursive
 fantasy
 monospace

font-size: xx-small winzig
 oder x-small sehr klein
 oder small klein
 oder medium mittel
 oder large groß
 oder x-large sehr-groß
 oder xx-large riesig
 oder smaller etwas kleiner als normal
 oder larger etwas größer als normal
 oder Fließkommazahl mit Einheit z.B. mm oder cm oder pt
 oder Prozent z.B. 3%

font-style: normal nicht kursiv
 oder vordefinierte z.B. italic; kursiv

font-variant: small-caps kleine Großbuchstaben (Kapitälchen)
 oder normal kein Kapitälchen

font-weight: bold fett
 oder bolder extra fett
 oder lighter dünn
 oder normal nicht dünn und nicht fett
 oder 100 extra dünn
 oder 200
 oder 300
 oder 400
 oder 500 medium
 oder 600
 oder 700 bold
 oder 800
 oder 900 extra fett



letter-spacing:	Abstand zwischen Zeichen im Text
	normal
oder	Fließkommazahl mit Einheit z.B. mm oder cm oder pt
line-height:	Abstand zwischen 2 Zeilen
	normal
oder	Fließkommazahl mit Einheit z.B. mm oder cm oder pt
text-align:	left Text linksbündig
oder	center Text zentriert
oder	right Text rechtsbündig
oder	justify Text im Blocksatz
text-decoration:	underline Text unterstrichen
oder	overline Text überstrichen
oder	line-through Text durchgestrichen
oder	blink Text blinkend
oder	none Text normal
text-indent:	Texteinschub
	Fließkommazahl mit Einheit z.B. mm oder cm oder pt
oder	Prozent z.B. 3%
text-transform:	capitalize Wortanfang ALLER Wörter auf Großbuchstabe
oder	uppercase alle Zeichen nach Großbuchstabe
oder	lowercase alle Zeichen nach Kleinbuchstabe
oder	none
white-space:	Zeilenumbruch ab IE 5.x
	normal automatischer Zeilenumbruch einschalten
oder	pre manuellen Zeilenumbruch einschalten
oder	nowrap kein Zeilenumbruch möglich
word-wrap:	Wortumbruch ab IE 5.x
	normal
oder	break-word

4.3.2.2.4.3.39.26.2. Farben- und Hintergrundeigenschaften

background:	Liste folgender Eigenschaften in fester Reihenfolge und Blanktrennung
	color wie background-color
	image wie background-image
	repeat wie background-repeat
	attachment wie background-attachment
	position wie background-position
oder	"null" transparenter Hintergrund
background-attachment:	scroll Hintergrundbild scrollt mit dem Objekt z.B. dem Dokument
oder	fixed Hintergrundbild ist fixiert
background-color:	Hintergrundfarbe
	#rrggbb
oder	rgb(rrr,ggg,bbb)
oder	vordefiniert
background-image:	none kein Hintergrundbild
oder	url("pfad_und_grafikdatei_name_als_zeichenkette")
background-repeat:	repeat Hintergrundbild horizontal und vertikal wiederholen
oder	no-repeat Hintergrundbild nicht horizontal und nicht vertikal wiederholen
oder	repeat-x Hintergrundbild horizontal wiederholen
oder	repeat-y Hintergrundbild vertikal wiederholen
color:	Textfarbe (Vordergrundfarbe)
	#rrggbb
oder	rgb(rrr,ggg,bbb)
oder	vordefiniert

4.3.2.2.4.3.39.26.3. Layouteigenschaften

border:	Rahmen ziehen auf allen 4 Seiten
	Eigenschaftensliste mit fester Reihenfolge mit Blanktrennung
	width wie border-width
	style wie border-style
	color wie border-color
border-top:	wie border aber für oberen Rahmenteil
border-bottom:	wie border aber für unteren Rahmenteil



border-left:	wie border aber für linken Rahmenteil
border-right:	wie border aber für rechten Rahmenteil
border-color:	Rahmenfarbe aller Rahmenteile #rrggbb oder rgb(rrr,ggg,bbb) oder vordefiniert
border-top-color:	wie border-color aber für oberen Rahmenteil
border-bottom-color:	wie border-color aber für unteren Rahmenteil
border-left-color:	wie border-color aber für linken Rahmenteil
border-right-color:	wie border-color aber für rechten Rahmenteil
border-style:	Stil aller Rahmenteile none kein rahmen oder dotted gepunktet oder dashed gestrichelt oder solid solid oder double doppelter oder groove 3D-Rahmen 1 oder ridge 3D-Rahmen 2 oder inset 3D-Rahmen 3 oder outset 3D-Rahmen 4
border-bottom-style:	wie border-style aber für unteren Rahmenteil
border-left-style:	wie border-style aber für linken Rahmenteil
border-right-style:	wie border-style aber für rechten Rahmenteil
border-width:	Dicke aller Rahmenteile medium oder thin oder thick oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
border-top-width:	wie border-width aber für oberen Rahmenteil
border-bottom-width:	wie border-width aber für unteren Rahmenteil
border-left-width:	wie border-width aber für linken Rahmenteil
border-right-width:	wie border-width aber für rechten Rahmenteil
margin:	Randabstand auf allen 4 Seiten auto oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm oder Prozentangabe
margin-top:	wie margin aber für oberen Abstand
margin-bottom:	wie margin aber für unteren Abstand
margin-left:	wie margin aber für linken Abstand
margin-right:	wie margin aber für rechten Abstand
padding:	Abstand Objekthalt zum Objektrand auf allen 4 Seiten auto oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm oder Prozentangabe
padding-top:	wie padding aber für oberen Abstand
padding-bottom:	wie padding aber für unteren Abstand
padding-left:	wie padding aber für linken Abstand
padding-right:	wie padding aber für rechten Abstand
scrollbar3d-light-color:	Scrollbar-Farbe bei 3D #rrggbb oder rgb(rrr,ggg,bbb) oder vordefiniert
scrollbar-arrow-color:	Scrollbar-Pfeile-Farbe ab IE 5.x #rrggbb oder rgb(rrr,ggg,bbb) oder vordefiniert
scrollbar-base-color:	Scrollbar-Basis-Farbe ab IE 5.x #rrggbb oder rgb(rrr,ggg,bbb) oder vordefiniert
scrollbar-dark-shadow-color:	Scrollbar-Farbe des dunklen Schattens ab IE 5.x #rrggbb oder rgb(rrr,ggg,bbb) oder vordefiniert
scrollbar-face-color:	Scrollbar-Face-Farbe ab IE 5.x #rrggbb oder rgb(rrr,ggg,bbb)



scrollbar-highlight-color: oder vordefiniert
 Scrollbar-Highlight-Farbe ab IE 5.x
 #rrggbb
 oder rgb(rrr,ggg,bbb)
 oder vordefiniert
 scrollbar-shadow-color: Scrollbar-Farbe des Schattens ab IE 5.x
 #rrggbb
 oder rgb(rrr,ggg,bbb)
 oder vordefiniert
 zoom: Objekt vergrößern ab IE 5.x
 normal
 oder Fließkommazahl als Faktor (1,0 ist normal)
 oder Prozentangabe mit % z.B. 50% (100% ist normal)

4.3.2.2.4.3.39.26.4. Positionierungsangaben

bottom: Abstand zum oben umgebenden Objekt ab IE 5.x
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 display Sichtbarkeit
 none unsichtbar
 block anzeigen
 inline anzeigen
 left: Abstand zum links umgebenden Objekt ab IE 5.x
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 beachte Wert der Eigenschaft document.all.position
 top: Abstand zum unten umgebenden Objekt
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 beachte Wert der Eigenschaft document.all.position
 right: Abstand zum rechts umgebenden Objekt ab IE 5.x
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 offset-left: wie left
 offset-top: wie top
 pixel-left: absolute X-Position der linken oberen Objekt-Ecke im Pixelsystem,
 dessen Ursprung (0,0) links oben liegt
 also Abstand vom linken Fensterrand
 nur Integerzahl ohne Einheit, da automatisch in Pixel interpretiert
 pixel-top: absolute Y-Position der linken oberen Objekt-Ecke im Pixelsystem,
 dessen Ursprung (0,0) links oben liegt
 also Abstand vom oberen Fensterrand
 nur Integerzahl ohne Einheit, da automatisch in Pixel interpretiert
 height: Höhe des Objektes
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 Hinweis: wenn geändert wird, so automatischer Umbruch des
 Inhaltes des HTML-Elementes
 width: Breite des Objektes
 auto
 oder Fließkommazahl mit Einheitenangabe z.B. pt oder cm
 oder Prozentangabe
 Hinweis: wenn geändert wird, so automatischer Umbruch des
 Inhaltes des HTML-Elementes
 visibility : Objektsichtbarkeit
 visible sichtbar
 oder hidden unsichtbar
 oder inherit Sichtbarkeit laut Elternobjekt
 z-index: auto jede neu erzeugte Objekt ist das oberstes



Achtung: sollte die Dimension des neuen Objektes sich nach der Erzeugung derart verändern, so dass es ein anderes Objekt überlagert, dann ändert sich trotzdem nichts an der Sichtbarkeit: da neue Objekt liegt unter dem vorhandenen Objekt, das vom neuen Objekt nun überlagert wird, denn zum Zeitpunkt der Erzeugung des neuen Objektes war keine Überlagerung ---> anstelle auto immer z-index kodieren !!!!

oder Schichtnummer des Objektes bei sich irgendwann überlagernden Objekten (also auch bei Überlagerung, die erst nach deren Erzeugung erfolgt)

ganzzahlig, also auch < 0
je kleiner um so tiefer in der Schicht
je höher um so höher in der Schicht
Sichtbar ist immer die oberste, also höchste Schicht
wenn 2 Objekte mit identischer Schichtnummer, so Sichtbarkeit laut Reihenfolge der Kodierung im Quelltext

4.3.2.2.4.3.39.26.5. Druckeigenschaften

page-break-after: wenn Druck des Objektes vollzogen, so soll Seitenumbruch erfolgen

	always	ja
oder	auto	Browser entscheidet
oder	""	nein

page-break-before: bevor der Druck des Objektes beginnt, so soll Seitenumbruch erfolgen

	always	ja
oder	auto	Browser entscheidet
oder	""	nein

4.3.2.2.4.3.39.26.6. Cursor

cursor:

	auto	
oder	crosshair (Kreuz)	
oder	default	je nach Windowseinstellung
oder	hand	
oder	move	Verschieben-Symbol in alle Richtungen
oder	N-resize	Verschiebe-Symbol in Richtung oben
oder	NE-resize	Verschiebe-Symbol in Richtung oben-rechts
oder	NW-resize	Verschiebe-Symbol in Richtung oben-links
oder	S-resize	Verschiebe-Symbol in Richtung unten
oder	SE-resize	Verschiebe-Symbol in Richtung unten-rechts
oder	SW-resize	Verschiebe-Symbol in Richtung unten-links
oder	E-resize	Verschiebe-Symbol in Richtung rechts
oder	W-resize	Verschiebe-Symbol in Richtung links
oder	text	Strich
oder	wait	Sanduhr
oder	help	Fragezeichen

Achtung: unter Windows sind Cursor-Belegungen veränderbar !!!
Bsp: anstelle Sanduhr ein anderes Symbol, dass dann natürlich auch bei wait erscheint

4.3.2.2.4.3.39.26.7. Bildausschnitt

clip enthält die Form UND Dimension des Bildausschnittes auf der Ebene des HTML-Elementes
wird eigentlich nur zum Schreiben verwendet
Form des Bildausschnittes: RECHTECKIG
Bsp. zu schreiben:

```
.clip=rect(x1,y1,x2,y2);
```

x1,y1	linke obere Ecke	x1	horizontal in Pixel
		y1	vertikal in Pixel
x2,y2	rechte untere Ecke	x2	horizontal in Pixel
		y2	vertikal in Pixel

x1 bis y2 immer bezüglich links oben im HTML-Element bzw. im Fenster des Browsers
können den Wert auto haben, der immer die maximale Dimension bewirkt
Bsp. 'rect:(auto,y1,auto,y2)';
wenn alle auf auto, so wird der Bildausschnitt maximiert, was der Standardimension



entspricht

4.3.2.2.4.3.40. styleSheet Objekt des Internet Explorer

ab IE 4.x

Ansatz:

Dieses Objekt dient der Verwaltung von Styles (StyleSheets) im Dokument, die auf ganz bestimmte Arten in das Dokument implementiert wurden:

Beispiele für Arten: <HEAD>
 <STYLE>
 @import url("MeineStyleSheetDatei.css");
 </STYLE>
 <HEAD>

 <LINK REL=stylesheet HREF="styles.css" type="text/css">

 per Pseudoklasse @page (Objekt page)

 <HEAD>
 <STYLE>
 P {color:green}
 </STYLE>
 </HEAD>

 per document.createStyleSheet('styles.css');

Diese o.g. Arten von Implementationen können als Block von Styles angesehen werden, mit anderen Worten: Als Menge von Style-Regeln.

Besonders für den Programmierer ist die vereinfachte Kodierung von Styles in externen CSS-Dateien und die Verwendung der browsereigenen Pseudoklassen wie @page und @import interessant: Anhand dieser sind komplette Style-Regeln-Manipulationen zur Laufzeit des HTML-Dokumentes möglich. So kann z.B. das Layout des ganzen HTML-Dokumentes nach seinem kompletten Laden auf einen Schlag geändert werden.

Allen Arten von Style-Implementationen haben eines gemeinsam: Es werden einzelne Styles im Browser verwaltet, die auch alle einzeln durch die übliche Style-Eigenschaften und Style-Methoden verwaltet werden könnten (siehe style Objekt).

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem

Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsen Bezeichners verwendet !!!

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

Verwaltung des styleSheet-Objektes in der Collection document.styleSheets:

Ein styleSheet-Objekt ist ein Element der Collection document.styleSheets, die in der Reihenfolge der Style-Implementationen im Quellcode der HTML-Seite beim Laden des Dokumentes gefüllt, also initialisiert wird. Jede o.g. Implementation erzeugt je ein styleSheet-Objekt als Element der Collection document.styleSheets.

Beispiel:

```
<HEAD>
<STYLE>
    .StyleRegel1 {color:"red"}
    .StyleRegel2 {color:"blue"}
</STYLE>
</HEAD>
```

Es wird ein styleSheet-Objekt erzeugt ,das 2 Regeln besitzt.

Verwaltung der Implementationsarten im styleSheet-Objekt:

Im styleSheet-Objekt wird jede o.g. Art der Style-Implementation einzeln verwaltet und zwar mit je einer eigenen Collection des styleSheet-Objektes. Ein styleSheet-Objekt referenziert selbst diese Collections, die je eine vordefinierte Art der Style-Implementation in das Dokument verwalten. Ob diese Collections auch reale Elemente besitzen, hängt davon ab, ob Styles in o.g. Arten implementiert wurden oder noch werden.

Collectionen des styleSheet Objektes Beispiele zu den Arten der verwalteten Style-Implementationen

```

styleSheet.imports      <HEAD>
                        <STYLE>
                        @import url("MeineStyleSheetDatei.css");
                        </STYLE>
                        <HEAD>

                        <LINK REL=stylesheet HREF="styles.css" type="text/css">

                        document.createStyleSheet('styles.css');

                        Methode .addImport()

styleSheet.pages        per Pseudoklasse @page (Objekt page)

styleSheet.rules        <HEAD>
                        <STYLE>
                        P {color:green}
                        </STYLE>
                        </HEAD>

```

Damit weist ein Eintrag in der Collection document.styleSheets auf ein styleSheet Objekt, das wiederum genau die zur Art der Implementation passende Collection referenziert, die wiederum konkrete Style-Objekte referenziert, welche eben nur auf eine von o.g. Arten der Implementationen in das Dokument eingebettet wurden und sich ansonsten **nicht** z.B. von einem inline-Style (per STYLE-Attribut im HTML-Tag eingebetteter Style) unterscheiden.

Beispiel:

```

<HEAD>
<STYLE>
    .StyleRegel1 {color:"red"}
    .StyleRegel2 {color:"blue"}
</STYLE>
<HEAD>

```

Implementation füllt Collection styleSheet.rules mit 2 Elementen (2 Regeln)

Wenn obige Implementation die ERSTE im Dokument ist, so wird das ERSTE styleSheet-Objekt erzeugt, also das Element document.styleSheets[0], das auf die Collection styleSheet.rules verweist.

Zugriff auf styleSheet-Objekt:

nur per Collection document.styleSheets

Hinweis: Style, der per Link oder @import anhand einer CSS-Datei eingebunden, kann zur Laufzeit **nur** verändert werden, wenn document.designMode auf "On" gesetzt ist

Beispiel 1:

```

<HEAD>
<STYLE>
    BODY {background-color: #CFCFCF;}
    @import url("MeineStyleSheetDatei.css");
</STYLE>
<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        // Style des BODY laut Klassendefinition
        var StyleSheetObjekt=document.styleSheets[0];

        // erste Regel zum Style des BODY holen
        var RegelObjekt = StyleSheetObjekt.rules[0];

        // und Style ändern
        RegelObjekt.style.backgroundColor="#0000FF";

        // neue Regel für BODY hinzufügen
        StyleSheetObjekt.addRule("BODY"," border-color: #FFFF00;");

        // und weitere neue Regel für BODY
        StyleSheetObjekt.imports[0].color="#000000";
    }
</SCRIPT>
</HEAD>

```

Beispiel 2:



```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeige(RegelIndex)
    {
        alert(      "Regel Nr " + RegelIndex
                    + " hat style.color = "
                    + document.styleSheets[0].rules.item(RegelIndex).style.color
                    + ".");
    }
</SCRIPT>
<STYLE>
    .StyleRegel1 { color:"red" }
    .StyleRegel2 { color:"blue" }
</STYLE>
</HEAD>
<BODY>
    <P CLASS=" StyleRegel1">
        StyleRegel1
    </P>
    <P CLASS=" StyleRegel2">
        StyleRegel2
    </P>

    <BUTTON onclick="Anzeige(0)">StyleRegel1</BUTTON>
    <BUTTON onclick="Anzeige(1)"> StyleRegel2</BUTTON>

</BODY>
</HTML>

```

Zugriff auf Collectionen des styleSheet-Objektes:

siehe Beschreibung der Collectionen

Eigenschaften des styleSheet-Objektes:

.href	Url einer externen Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
.owningElement	Referenz auf den im StyleSheet implementierten Style als Kindobjekt
	Hinweis: Es sind die Eigenschaften und Methoden des style Objektes referenzierbar

Beispiel:

```

for ( var i = 0; i < document.styleSheets.length; i++ )
{
    if ( document.styleSheets(i).owningElement.tagName == "STYLE" )
    {
        for ( var j = 0; j < document.styleSheets(i).imports.length; j++ )
        {
            alert(      "Importierter StyleSheet (Style-Regel) Nr " + j
                        + " hat HREF = " + document.styleSheets(i).imports(j).href
                        );
        }
    }
}

```

.parentStyleSheet	Referenz auf das den StyleSheet implementierende Objekt (Elternobjekt)
.readOnly	Art der Implementation des StyleSheets im Dokument ermitteln
.title	Titel des StyleSheets
.type	Mimetypt des StyleSheets muss "text/css" sein

Methoden des styleSheet-Objektes:

.addImport()	StyleSheet aus externer CSS-Datei importieren und als Element der Collection styleSheet.imports erzeugen (entspricht @import url() innerhalb STYLE im HEAD)
.addPageRule()	StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.pages erzeugen (entspricht @page vom Objekt page)

Beispiel:

```

function TextFaerben ()
{document.styleSheets[0].addPageRule("DIV B", "color:blue", 0);}

<DIV onmouseover="TextFaerben ();">
    <B>dieser Text wird per CSS mit blauer Fabrbe angezeigt</B>
</DIV>

```

.addRule()	StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.rules erzeugen (entspricht xx { ...} innerhalb STYLE im HEAD)
------------	---

Beispiel:

```

function TextFaerben ()
{document.styleSheets[0].addRule("DIV B", "color:blue", 0);}

```



```

    <DIV onmouseover="TextFärben ();">
        <B>dieser Text wird per CSS mit blauer Farbe angezeigt</B>
    </DIV>
    .fireEvent()      ein Event auslösen
                      true      Event erfolgreich ausgelöst
                      false     Event nicht ausgelöst
    .removeRule()     StyleSheet aus der Collection styleSheet.rules entfernen
                      Achtung: Um Style auch sichtbar zu entfernen, muss
                              Dokument neu geladen werden
                              oder      alle betroffene Style-Elemente neu mit Wert belegen
                                      durch Zuweisung auf sich selbst (siehe Beispiel)

```

Beispiel:

```

<STYLE>
    P { color:green }
</STYLE>
<SCRIPT>
    function StyleEntfernen()
    {
        var StyleSheetsObjekt = document.styleSheets;
        var AnzahlStyleSheets = StyleSheetsObjekt.length;

        if (AnzahlStyleSheets > 0)
        {
            // StyleSheet mit Index 0 bearbeiten also P {color:green}
            var StyleSheetAnIndex0 = StyleSheetsObjekt[0];

            // wobei P {color:green} eine Regel ist, also Collection rules verwenden
            var StyleSheetsRegelCollection = StyleSheetAnIndex0.rules;

            var AnzahlRegeln = StyleSheetsRegelCollection.length;

            if (AnzahlRegeln > 0)
            {
                // Regel P {color:green} entfernen
                StyleSheetAnIndex0.removeRule(0);

                // visuell auch die Farbe entfernen durch Zuweisung auf sich selbst also
                // nun ohne die Regel P {color:green}
                ID_P.innerHTML= ID_P.innerHTML;
            }
        }
    }
</SCRIPT>

<P ID="ID_P" >Test</P>
<BUTTON onclick="StyleEntfernen()">Text im P-Tag entfärben.</BUTTON>

```

4.3.2.2.4.3.40.1. **styleSheet.imports Collection des Internet Explorer**

verwaltet die per

Link auf eine externe Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
 @import einer externen Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
 Methode .addImport()

importierten Style-Regeln

```

z.B.    <HEAD>
        <STYLE>
            @import url("MeineStyleSheetDatei.css");
        </STYLE>
    </HEAD>

z.B.    <LINK REL=stylesheet HREF="styles.css" type="text/css">

z.B.    document.createStyleSheet('styles.css');

```

Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Syntax:

```

[ var FeldZeiger = ] document.styleSheets[Index1].imports
[ var FeldElementZeiger = ] document.styleSheets[Index1].imports[Index2]

```

Index1: Integer und ab 0
muss in [] kodiert sein

Index2: Integer und ab 0
muss in [] kodiert sein



Beispiel:

```
for ( var i = 0; i < document.styleSheets.length; i++ )
{
    if ( document.styleSheets(i).owningElement.tagName == "STYLE" )
    {
        for ( var j = 0; j < document.styleSheets(i).imports.length; j++ )
        {
            alert(      "Importierter StyleSheet (Style-Regel) Nr " + j
                        + " hat HREF = " + document.styleSheets(i).imports(j).href
                        );
        }
    }
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.3.40.2. page Objekt des Internet Explorer

repräsentiert eine @page Regel in einem styleSheet Objekt
ab IE 5.5

@page Regel:

Regel für Dimensionen, Orientierungen und Margins in einer Seite per styleSheet Objekt

Regel ist eine Pseudoklasse von Style

Seite wird als Rechteckbereich aufgefasst, der eingeteilt ist in:

Seitenbereich: Inhalt der Seite z.B. Text, Grafik

Marginbereich: Rand um den Seitenbereich

Syntax:

HTML @page Kette1 { Kette2 }

Kette1	":first"	Regel gehört der 1. Seite
	":footer"	Regel gehört der Fußnote
	":header"	Regel gehört der Kopfnote
	":left"	Regel gehört der linken Seite
	":left : header"	Regel gehört der Kopfnote Seite links
	":left : footer"	Regel gehört der Fußnote Seite links
	":right"	Regel gehört der rechten Seite
	":right:header"	Regel gehört der Kopfnote Seite rechts
	":right:footer"	Regel gehört der Fußnote Seite rechts

Kette2 String aus Regelfolge mit Semikolontrennung

Beispiel: @page : left { font-weight:bold;font_style:italic; }

Eigenschaften:

.pseudoClass Pseudoklasse einer Seite oder @page Regel per Objekt page

.selector Seiten-Selektor per Objekt page

Methoden:

keine

4.3.2.2.4.3.40.3. styleSheet.pages Collection des Internet Explorer

verwaltet die per Pseudoklasse @page (Objekt page) importierten Style-Regeln

Syntax:

```
[ var FeldZeiger = ] document.styleSheets[Index1].pages
[ var FeldElementZeiger = ] document.styleSheets[Index1].pages[Index2]
```

Index1: Integer und ab 0
 muss in [] kodiert sein

Index2: Integer und ab 0
 muss in [] kodiert sein

Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>

4.3.2.2.4.3.40.4. styleSheet.rules Collection des Internet Explorer

verwalten die per



xxx { ... } innerhalb von STYLE im HEAD
oder per .addRule()

importierten Style-Regeln

z.B. <HEAD>
 <STYLE>
 P {color:green}
 </STYLE>
 </HEAD>

Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Syntax:

```
[ var FeldZeiger = ] document.styleSheets[Index1].rules
[ var FeldElementZeiger = ] document.styleSheets[Index1].rules[Index2]
```

Index1: Integer und ab 0
 muss in [] kodiert sein

Index2: Integer und ab 0
 muss in [] kodiert sein

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function Anzeige(RegelIndex)
{
    alert(      "Regel Nr " + RegelIndex
               + " hat style.color = "
               + document.styleSheets[0].rules.item(RegelIndex).style.color
               + ".");
}
</SCRIPT>
<STYLE>
.StyleRegel1 {color:"red"}
.StyleRegel2 {color:"blue"}
</STYLE>
</HEAD>
<BODY>
<P CLASS=" StyleRegel1">
    StyleRegel1
</P>
<P CLASS=" StyleRegel2">
    StyleRegel2
</P>

<BUTTON onclick=" Anzeige(0)">StyleRegel1</BUTTON>
<BUTTON onclick=" Anzeige(1)">StyleRegel2</BUTTON>
</BODY>
</HTML>
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des

4.3.2.2.4.3.41. document.styleSheets Collection des Internet Explorer

Feld der Zeiger aller styleSheet Objekte im Dokument (siehe dort)

Syntax:

```
[ var ZeigerAufFeld = ] document.styleSheets
[ var ZeigerAufFeldElement = ] document.styleSheets [Index [, SubIndex]]
```

Index Integer ab 0
oder String Name oder ID des Elementes
 muss in [] kodiert sein

SubIndex optional
 nur kodieren wenn Index ein String ist
 Integer als Unterindex also Unterelement eines Elementes

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem
Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)



Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

Eigenschaften:

.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.sourceIndex	Index des Objektes in der Collection document.all

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.42. table Objekt des Internet Explorer

Objekt einer Tabelle (HTML-Element TABLE)

4.3.2.2.4.3.42.1. Erzeugung der Tabelle

4.3.2.2.4.3.42.1.1. Erzeugung in HTML

Die Tabelle kann folgende Tabellenelemente (Tags) besitzen:

CAPTION,
COL,
COLGROUP,
TBODY,
TD,
TFOOT,
TH,
THEAD
TR

wobei in der Tabelle maximal 1 THEAD
1 TFOOT
1 CAPTION (Überschrift)

auftauchen kann

TBODY-Tag nicht kodiert werden muss, wenn keine Fehlzuordnung zu THEAD **und** kein TFOOT möglich ist

Beispiel:

```
<TABLE BORDER=1 WIDTH=80% BGCOLOR="gray">
<THEAD BGCOLOR="blue">
  <TR>
    <TH>Titel Spalte 1</TH>
    <TH>Titel Spalte 2</TH>
  </TR>
</THEAD>
<TBODY BGCOLOR="yellow">
  <TR>
    <TD>Zeile 1, Spalte 1 </TD>
    <TD>Zeile 1, Spalte 2 </TD>
  </TR>
  <TR>
    <TD>Zeile 2, Spalte 1</TD>
    <TD>zeile 2, Spalte 2</TD>
  </TR>
</TBODY>
```




```

<TFOOT BGCOLOR="green">
  <TR>
    <TD COLSPAN="4">TFOOT</TD>
  </TR>
</TFOOT>
<CAPTION VALIGN="BOTTOM" STYLE="font-size:10;">
  Caption
</CAPTION>
</TABLE>

```

Tabelle wird erst angezeigt, wenn das Dokument komplett geladen ist und das Event onload ausgelöst wurden.

4.3.2.2.4.3.42.1.2. Erzeugung in JScript

IE 4.x

Die Erzeugung und Manipulation der Tabelle sind in Script erst möglich, wenn das Dokument komplett geladen und das Event onload ausgelöst wurden. Veränderungen der Tabellenelemente per Style werden sofort sichtbar. Veränderungen der Tabelle bezüglich Art und Anzahl der Tabellenelemente werden z.T. erst nach Aufruf der Methode .refresh() sichtbar.

4.3.2.2.4.3.42.2. Daten der Tabelle

4.3.2.2.4.3.42.2.1. Datenbereitstellung

4.3.2.2.4.3.42.2.1.1. in HTML

Die Daten sind bereits in der Kodierung der Tabelle implementiert und damit von statischer Natur.

Die Berechnung des Layouts der Tabelle erfolgt einmalig.

Alternativ erfolgt die Erzeugung eines HTML-Codes mit Daten z.B. per PHP auf Basis eines Datenbankservers.

4.3.2.2.4.3.42.2.1.2. in JScript

Die Daten sind während und nach der Erzeugung der Tabelle manipulierbar und damit von dynamischer Natur.

Die Berechnung des Layouts der Tabelle erfolgt automatisch .

4.3.2.2.4.3.42.2.1.3. per Active-X-Control

Für den Internet Explorer existiert eine dynamische Datenbereitstellung über das ActiveX-Control TDC (Tabular Data Container) mit dem Class-ID "clsid:333C7BC4-460F-11D0-BC04-0080C7055A83" .

TDC liest eine Textdatei, deren Inhalt bestimmten Regeln entsprechen muss, und stellt die Daten aus der Textdatei in der Tabelle dar.

Die Verwaltung der Textdatei, also deren Daten, ist sehr einfach.

Nachteilig beim TDC ist, dass die Textdatei im Browser-Cache liegen muss und damit vom User manipulierbar ist. TDC unterstützt keine Verschlüsselung der Text-Datei.

TDC wird an anderer Stelle in dieser Dokumentation beschrieben. Leider ist TDC kein HTML-Standard.

Das Tabellen-Objekt besitzt bereits Eigenschaften und Methoden, die vom TDC benutzt werden.

4.3.2.2.4.3.42.2.2. Dateneinbindung

4.3.2.2.4.3.42.2.2.1. in HTML

Die Daten sind bereits in der Kodierung der Tabelle implementiert und damit einmalig eingebunden. Alternativ erfolgt die Erzeugung eines HTML-Codes z.B. per PHP auf Basis eines Datenbankservers.

4.3.2.2.4.3.42.2.2.2. in JScript

Die Daten sind während und nach der Erzeugung der Tabelle manipulierbar.

Die Berechnung des Layouts der Tabelle erfolgt automatisch .

4.3.2.2.4.3.42.2.2.3. per Active-X-Control

Die Daten werden einmalig eingebunden.

4.3.2.2.4.3.42.3. Tabellen-Objektmodell (TOM) in JScript

Für die dynamische Verwaltung einer Tabelle wird ein eigenes Tabellenmodell (TOM) verwendet, welches aber mit dem HTML-DOM (DOM) kommuniziert. Tabellenelemente sind also im DOM und TOM verfügbar und müssen Eigenschaften und Methoden zum DOM und TOM implementiert haben.

Die Tabelle ist der Container (Eltern) für ihre Elemente.

Die Eigenschaften und Methoden der Tabelle im TOM werden in der Regel unter Beibehaltung des Bezeichners durch die Eigenschaften und Methoden des Tabellenelementes im TOM überschrieben und **nicht** vererbt.

Im TOM sind auch die Methoden implementiert, die das Füllen der Tabelle mit den Tabellenelemente (außer TBODY) zulassen.

Alle Tabellenlemente TBODY müssen

entweder in HTML kodiert sein

oder über die Methoden des DOM erzeugt werden.

TOM ist nicht in der Lage, TBODY-Elemente zu erzeugen.

Der Aufwand in der DHTML-Programmierung ist wesentlich höher dafür abstrakter als die pure HTML-Kodierung der Tabelle im Dokument. Aber letztere lässt keine dynamische Struktur- und Datenverwaltung zu.

4.3.2.2.4.3.42.4. Methoden des DOM und TOM zur Verwaltung der Tabellenelemente (Übersicht)

Zur physischen Veränderung der Tabelle und ihrer Elemente sind folgende Methoden im TOM und DOM implementiert:



Methoden der Tabelle TABLE als Objekt:

.createTHead()	THEAD erzeugen
.deleteTHead()	THEAD löschen
.createTFoot()	TFOOT erzeugen.
.deleteTFoot()	TFOOT löschen
.createCaption()	CAPTIOIN erzeugen.
.deleteCaption()	CAPTION löschen
.insertRow()	Zeile erzeugen
.deleteRow()	Zeile löschen
.moveRow()	Zeilen austauschen

Methoden des Tabellenkörpers TBODY als Objekt:

.insertRow()	Zeile erzeugen
.deleteTHead()	THEAD löschen
.deleteTFoot()	TFOOT löschen
.deleteRow()	Zeile löschen
.moveRow()	Zeilen austauschen

Methoden des Tabellenkopfes THEAD als Objekt:

.insertRow()	Zeile erzeugen
.deleteRow()	Zeile löschen
.moveRow()	Zeilen austauschen

Methoden des Tabellenfusses TFOOT als Objekt:

.insertRow()	Zeile erzeugen
.deleteRow()	Zeile löschen
.moveRow()	Zeilen austauschen

Methoden der Tabellenzeile TR als Objekt:

.deleteCell()	Zelle löschen in Zeile, Methode von TR
.insertCell()	Zelle erzeugen in Zeile, Methode von TR

Die Erzeugung von TBODY ist im TOM nicht implementiert. Dafür ist **nur** DOM zu verwenden.

Die Erzeugung /Löschung von Tabellenelementen bewirkt Änderung im DOM und in den betroffenen Collectionen des TOM.

4.3.2.2.4.3.42.5. **Dynamische Struktur und Daten einer Tabelle per JScript**

4.3.2.2.4.3.42.5.1. *Strukturerzeugung der Tabelle*

4.3.2.2.4.3.42.5.1.1. *in HTML*

Wird die HTML-Kodierung einer Tabelle im BODY-Teil des Dokumentes gewünscht, so ist zu beachten:

Im HTML-Code muss eine leere Tabelle aus folgenden Tags kodiert werden:

TABLE-Tag
allen TBODY-Tags

Für die Tags sind ID-Attribute zu kodieren.

Die Erzeugung der anderen Tabellenelemente erfolgt per Script, das aktiviert wird nach dem kompletten Laden des Dokumentes und damit dem Laden der leeren Tabelle (onload="..." kodieren im BODY-Tag)

Beispiel für Erzeugung einer Tabelle in Script per HTML, TOM und DOM:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    function TabelleFuellen()
    {
        var Zeile;
        var Zelle;

        // +++++ Tabellenrahmen und Rahmenfarbe +++++
        ID_Tabelle.border = 1;
        ID_Tabelle.bgColor = "magenta";

        // +++++ TabellenKopf füllen +++++
        // ----- erzeugen
        var TabellenKopf = ID_Tabelle.createTHead();

        // ----- Hintergrundfarbe
```



```

TabellenKopf.bgColor = "blue";

// ----- mit 1 Zeile
Zeile = TabellenKopf.insertRow();

// Zeile mit 1 Zellenfüllen
Zelle = Zeile.insertCell();
Zelle.align = "center";
Zelle.style.fontWeight = "bold";
Zelle.innerText = "Tabellenkopf Zelle";

// +++++ Tabellenbody 0 füllen +++++
// wurde per HTML-Kodierung im BODY erzeugt
// ----- Hintergrundfarbe
ID_TBody0.bgColor = "green";

//----- 2 Zeilen mit je 1 Zelle erzeugen
for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
{
    // ----- Zeile erzeugen
    Zeile = ID_TBody0.insertRow();

    // ----- Zelle in der Zeile erzeugen
    Zelle = Zeile.insertCell();

    // und füllen
    Zelle.innerText = "TBODY 0 Zeile " + ZeilenZahler + " Zelle"
}

// +++++ Tabellenbody 1 füllen +++++
// wurde per HTML-Kodierung im BODY erzeugt
// ----- Hintergrundfarbe
ID_TBody1.bgColor = "yellow";

//----- 2 Zeilen mit je 1 Zelle erzeugen
for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
{
    // ----- Zeile erzeugen
    Zeile = ID_TBody1.insertRow();

    // ----- Zelle in der Zeile erzeugen
    Zelle = Zeile.insertCell();

    // und füllen
    Zelle.innerText = "TBODY 1 Zeile " + ZeilenZahler + " Zelle"
}

// +++++ TabellenFuss füllen +++++
// ----- erzeugen
var TabellenFuss = ID_Tabelle.createTFoot();

// ----- mit Hintergrundfarbe
TabellenFuss.bgColor = "brown";

// ----- mit 1 Zeile
Zeile = TabellenFuss.insertRow();

// Zeile mit 1 Zellenfüllen
Zelle = Zeile.insertCell();
Zelle.align = "center";
Zelle.style.fontWeight = "bold";
Zelle.innerText = "Tabellenfuss Zelle";
Zelle.colSpan = "1";
Zelle.id = "ZelleImTabellenFuss";

// +++++ TabellenCapiton füllen +++++
// ----- erzeugen
var TabellenCaption = ID_Tabelle.createCaption();

// ----- und füllen
TabellenCaption.align = "bottom";
TabellenCaption.style.fontSize = "10";
TabellenCaption.innerText = "TabelleCaption"
}

```



```

        function ZelleImTabellenFussAendern()
        { ZelleImTabellenFuss.innerHTML = "Tabellenfuss Zelle neu" }
    </SCRIPT>
</HEAD>
<BODY onload="TabelleFuellen();">
    <!-- Tabelle leer mit allen TBODY erzeugen, aber komplett ohne Daten, da sonst nicht manipulierbar !!! /-->
    <TABLE ID="ID_Tabelle"
        BORDER
        BGCOLOR="gray"
    >
        <TBODY ID="ID_TBod0"></TBODY>
        <TBODY ID="ID_TBod1"></TBODY>
    </TABLE>
    <BR>
    Tabelle mit HTML und TOM erzeugt
    <BR>
    <BUTTON onclick="ZelleImTabellenFussAendern();">Zelle im Tabellenfuss aendern</BUTTON>
</BODY>
</HTML>

```

4.3.2.2.4.3.42.5.1.2. *per Methoden des DOM*

Wird die HTML-Kodierung einer Tabelle **nicht** im BODY-Teil des Dokumentes gewünscht, so ist zu beachten:

Es sind alle Elemente (inklusive der Tabelle) per Methode `.createElement()` leer zu erzeugen und dann per Methode `.appendChild()` in der richtigen Reihenfolge in das DOM einzubinden.

Es sollte im BODY-Teil des Dokumentes ein leerer HTML-Container z.B. DIV mit ID-Attribut kodiert sein. In diesen Container erfolgt die Einbindung der Tabelle und damit in das Dokument. Der Container kann frei positioniert sein (und damit die Tabelle). Alternativ ist auch das Einbinden direkt in den BODY-Teil möglich, wobei das ID-Attribut im BODY-Tag verwendet wird. Der BODY-Teil des Dokumentes kann aber nicht positioniert werden (abgesehen vom Scrollen des Dokumentes im Anzeigefenster).

Die Erzeugung **aller** Tabellenelemente erfolgt per Script, das aktiviert wird nach dem kompletten Laden des Dokumentes (onload="..." kodieren im BODY-Tag)

Beispiel für Erzeugung einer Tabelle in Script per DOM und TOM aber ohne HTML:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    function TabelleFuellen()
    {
        var Zeile;
        var Zelle;

        // +++++ Tabelle erzeugen +++++
        var Tabelle = document.createElement("TABLE");

        // ----- Tabellenrahmen und Rahmenfarbe
        Tabelle.border = 1;
        Tabelle.bgColor = "magenta";

        // +++++ Tabellenelemente erzeugen +++++
        var TabellenKopf = document.createElement("THEAD");
        var Tabellenbody0 = document.createElement("TBODY");
        var Tabellenbody1 = document.createElement("TBODY");
        var TabellenFuss = document.createElement("TFOOT");
        var TabellenCaption = document.createElement("CAPTION");

        // +++++ Tabelle zusammenbauen +++++
        Tabelle.appendChild(TabellenKopf);
        Tabelle.appendChild(Tabellenbody0);
        Tabelle.appendChild(Tabellenbody1);
        Tabelle.appendChild(TabellenFuss);
        Tabelle.appendChild(TabellenCaption);

        // +++++ TabellenKopf füllen +++++

        // ----- mit Hintergrundfarbe
        TabellenKopf.bgColor = "blue";

        // ----- mit 1 Zeile
        Zeile = document.createElement("TR"); // erzeugt automatisch Start- und Ende-Tag
        TabellenKopf.appendChild(Zeile);
    }

```



```

//      Zeile mit 1 Zellenfüllen
Zelle      = Zeile.insertCell();
Zelle.align      = "center";
Zelle.style.fontWeight = "bold";
Zelle.innerHTML      = "Tabellenkopf Zelle";

// +++++ Tabellenbody 0 füllen +++++
// ----- Hintergrundfarbe
Tabellebody0.bgColor = "green";

//----- 2 Zeilen mit je 1 Zelle erzeugen
for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
{
    // ----- Zeile erzeugen
    Zeile = document.createElement("TR"); // erzeugt automatisch Start- und Ende-Tag
    Tabellebody0.appendChild(Zeile);

    // ----- Zelle in der Zeile erzeugen
    Zelle = document.createElement("TD"); // erzeugt automatisch Start- und Ende-Tag
    Zeile.appendChild(Zelle);

    //      und füllen
    Zelle.innerHTML = "TBODY 0 Zeile " + ZeilenZahler + " Zelle"
}

// +++++ Tabellenbody 1 füllen +++++
// ----- Hintergrundfarbe
Tabellebody1.bgColor = "yellow";

//----- 2 Zeilen mit je 1 Zelle erzeugen
for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
{
    // ----- Zeile erzeugen
    Zeile = document.createElement("TR"); // erzeugt automatisch Start- und Ende-Tag
    Tabellebody1.appendChild(Zeile);

    // ----- Zelle in der Zeile erzeugen
    Zelle = document.createElement("TD"); // erzeugt automatisch Start- und Ende-Tag
    Zeile.appendChild(Zelle);

    //      und füllen
    Zelle.innerHTML = "TBODY 1 Zeile " + ZeilenZahler + " Zelle"
}

// +++++ TabellenFuss füllen +++++
// ----- mit Hintergrundfarbe
TabelleFuss.bgColor = "brown";

// ----- mit 1 Zeile
Zeile      = document.createElement("TR"); // erzeugt automatisch Start- und Ende-Tag
TabelleFuss.appendChild(Zeile);

//      Zeile mit 1 Zellenfüllen
Zelle      = document.createElement("TD"); // erzeugt automatisch Start- und Ende-Tag
Zeile.appendChild(Zelle);

Zelle.align      = "center";
Zelle.style.fontWeight = "bold";
Zelle.innerHTML      = "Tabellenfuss Zelle";
Zelle.colSpan      = "1";
Zelle.id      = "ZelleImTabellenFuss";

// +++++ TabellenCaption füllen +++++
TabelleCaption.align = "bottom";
TabelleCaption.style.fontSize = "10";
TabelleCaption.innerHTML = "TabelleCaption"

// +++++ Tabelle in den DIV einbinden +++++
ID_Div.appendChild(Tabelle);
}

```



```

        function ZelleImTabellenFussAendern()
        {ZelleImTabellenFuss.innerHTML = "Tabellenfuss Zelle neu"}
    </SCRIPT>
</HEAD>
<BODY onload="TabelleFuellen();">
    <!-- Tabelle wird im DIV erzeugt wobei das ID des DIV verwendet wird , um die Tabelle einzubinden --->
    <DIV ID="ID_Div"></DIV>
    <BR>
    Tabelle mit DOM und TOM erzeugt
    <BR>
    <BUTTON onclick="ZelleImTabellenFussAendern();">Zelle im Tabellenfuss aendern</BUTTON>
</BODY>
</HTML>

```

4.3.2.2.4.3.42.5.2. Datenerzeugung mit der Strukturbildung und zur Laufzeit

Die dynamische Struktur muss per Script erzeugt worden sein. Bei der Erzeugung der Struktur können bereits Daten implementiert werden.

Die dynamische Änderung von Daten einer Tabelle kann zur Laufzeit nur durch den Bezug auf die jeweilige Zelle in der Zeile erfolgen.

Folgende Eigenschaften müssen daher für eine Zelle in der Struktur kodiert worden sein:

.id	für den Bezug auf die Zelle immer kodieren alternativ: Zugriff über die Collectionen rows und cells
.innerText	für den Textinhalt einer Zelle (Plain-Text) alternativ auch .innerHTML
.innerHTML	für den HTML-Inhalt einer Zelle alternativ auch .innerText immer notwendig zur Erzeugung von TH

Nur bei einer Zelle sind die Eigenschaften .innerHTML bzw. .innerText. schreibbar.

Beispiel Kalender:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
// Kalender aktueller Monat
// nur für IE

// ##### änderbare Variablen #####

// +++++ Position des Kalenders +++++
var Kalender_Top=20;
var Kalender_Left=20;

// +++++ Farben und Schrift-Layout +++++

// Schriften: Es sollten nur Fonts verwendet werden, die auf jedem Windows-PC installiert sind

// ----- Kalenderkopf -----

var Kalender_Kopf_SchriftArt='Times New Roman, Arial';
var Kalender_Kopf_SchriftGroesse= 5;
var Kalender_Kopf_SchriftFarbe='#CCEEFF';
var Kalender_Kopf_HintergrundFarbe='#3A6EA5'; // identisch mit BGColor des BODY
var Kalender_SpaltenKopf_SchriftFarbe='#AAFFFF';

// ----- Tag -----

var Tag_SchriftArt='Times New Roman, Verdana, Arial';
var Tag_SchriftGroesse=4;
var Tag_SchriftFarbe='#CCCCCC';
var Tag_HintergrundFarbe=Kalender_Kopf_HintergrundFarbe;
var Tag_Sonntag_SchriftFarbe='#FFDD00';
var Tag_Heute_SchriftFarbe='#FFFFFF';
var Tag_Heute_HintergrundFarbe='#1A4E85';
var Tag_Heute_Sonntag_SchriftFarbe=Tag_Heute_SchriftFarbe;
var Tag_Heute_Sonntag_HintergrundFarbe=Tag_Heute_HintergrundFarbe;

// +++++ Timer +++++

// Timer für Kalender zur Erkennung von Tages-, Monats- und Jahreswechsel verwenden

```



```

var Kalender_Timer_Verwenden=false; // true für verwenden, false für nicht verwenden

// Timer in Millisekunden für Aktualisierung des Kalenders
// nur verwendet wenn Kalender_Timer_Verwenden auf true
// Wert so hoch wie möglich setzen damit durch den Timer wenig Ressourcen genutzt werden
var Kalender_TimerWert=5000;

// +++++ Monatsnamen +++++
+++++

var MonatsNamenFeld = new Array
(
    'Januar',
    'Februar',
    'M&auml;r',
    'April',
    'Mai',
    'Juni',
    'Juli',
    'August',
    'September',
    'Oktober',
    'November',
    'Dezember'
);

// +++++ Tagnamen kurz +++++
+++++

var TagKurzNamenFeld = new Array // Woche beginnt mit Montag
(
    'Mo',
    'Di',
    'Mi',
    'Do',
    'Fr',
    'Sa',
    'So'
);

// ##### interne Variablen #####
// alle Variablen möglichst Wert-Typ-gerecht initialisieren (bei Zeiger kein "nil" oder
// "null" verwenden, also Zeiger ohne init).

var TagKurzNamenFeld_Laenge=TagKurzNamenFeld.length;

// Zeit- und Datum-Berechnungen
var Zeit_Jetzt;
var Zeit_Jetzt_Kette="";
var Zeit_Jetzt_Tag_Numerisch=0;
var Zeit_Jetzt_Monat_Numerisch=0;
var Zeit_Jetzt_Jahr_Numerisch=0;
var Zeit_Jetzt_Stunden_Numerisch=0;
var Zeit_Jetzt_Minuten_Numerisch=0;
var Zeit_Jetzt_Sekunden_Numerisch=0;
var Zeit_Vormonat;
var Zeit_VormonatErsterTag;

// Kalender-Anzeige: Globale Variablen, da Anzeige per Timer gesteuert werden kann
// und somit nicht pro Aufruf die Variablen neu erzeugt werden müssen.
var Anzeige_Kalender_TagesZahler=0;
var Anzeige_Kalender_HTMLCode="";
var Anzeige_Kalender_Kette="";
var Anzeige_Kalender_Zahler_TR=0;
var Anzeige_Kalender_Zahler_TD=0;
var Anzeige_Kalender_Tag_Kette1="";
var Anzeige_Kalender_Tag_Kette2="";
var Anzeige_Kalender_Tag_Kette3="";
var Anzeige_Kalender_Sonntag_Kette1="";
var Anzeige_Kalender_Sonntag_Kette2="";
var Anzeige_Kalender_Sonntag_Kette3="";

// ##### Funktionen der Zeit- und Datum-Berechnungen #####
+++++

function Zeit_AktuelleAngabenHolen()

```



```

{
    Zeit_Jetzt          = new Date();
    Zeit_Jetzt_Tag_Numerisch = Zeit_Jetzt.getDate();
    Zeit_Jetzt_Monat_Numerisch = Zeit_Jetzt.getMonth() + 1;
    Zeit_Jetzt_Jahr_Numerisch = Zeit_Jetzt.getYear();
    Zeit_Jetzt_Stunden_Numerisch = Zeit_Jetzt.getHours();
    Zeit_Jetzt_Minuten_Numerisch = Zeit_Jetzt.getMinutes();
    Zeit_Jetzt_Sekunden_Numerisch = Zeit_Jetzt.getSeconds();
    Zeit_Vormonat        = new Date(Zeit_Jetzt_Jahr_Numerisch, Zeit_Jetzt_Monat_Numerisch-1, 1);
    Zeit_VormonatErsterTag = Zeit_Vormonat.getDay();

    if(Zeit_Jetzt_Jahr_Numerisch < 100)
    {Zeit_Jetzt_Jahr_Numerisch+=1900;}

    if(Zeit_Jetzt_Jahr_Numerisch < 100)
    {Zeit_Jetzt_Jahr_Numerisch+=1900;}
}

function Zeit_NumerischNachString_MitVornull(NumerischerWert)
{
    Zeit_Jetzt_Kette=NumerischerWert.toString();

    // auf Vornull prüfen
    if (NumerischerWert < 10)
    {Zeit_Jetzt_Kette='0' + Zeit_Jetzt_Kette;}

    return Zeit_Jetzt_Kette;
}

// ##### Funktionen des Kalenders #####

// +++++ TR erzeugen +++++

function Anzeige_Kalender_HTMLCode_TR_Erzeugen(
    FarbeBackground,
    BreiteAlsString,
    HoeheAlsString,
    Ausrichtung_Horizontal,
    Ausrichtung_Vertikal,
    SpaltenSpannweiteAlsString,
    SchriftGroesse,
    SchriftFarbe,
    SchriftArt,
    Kette
)

// erweitert Anzeige_Kalender_HTMLCode
{
    // +++++ TR erzeugen: Beginn-Tag
    Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + '<TR>';

    // +++++ TD (Zelle) erzeugen

    // ----- Tag-Beginn
    Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + '<TD>';

    // ----- Attribut Background
    if (FarbeBackground != "")
    {Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' BGCOLOR="' + FarbeBackground + '"';}

    // ----- Attribut Breite
    if (BreiteAlsString != "")
    {Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' WIDTH=' + BreiteAlsString;}

    // ----- Attribut Höhe
    if (HoeheAlsString != "")
    {Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' HEIGHT=' + HoeheAlsString;}

    // ----- Attribut Ausrichtung horizontal
    if (Ausrichtung_Horizontal != "")
    {Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' ALIGN=' + Ausrichtung_Horizontal;}

    // ----- Attribut Ausrichtung vertikal
    if (Ausrichtung_Vertikal != "")
    {Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' VALIGN=' + Ausrichtung_Vertikal;}
}

```




```

// ---- Attribut Spannweite
if (SpaltenSpannweiteAlsString != "")
{Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + ' COLSPAN=' + SpaltenSpannweiteAlsString;}

// ---- Tag-Ende
Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + '>';

// ---- Inhalt der Zelle als Font mit Text
Anzeige_Kalender_HTMLCode=
    Anzeige_Kalender_HTMLCode
    + '<FONT SIZE=' + SchriftGroesse
    + ' COLOR=' + SchriftFarbe
    + ' FACE="' + SchriftArt + '"'
    + '>'
    + Kette
    + '</FONT>'

// ---- Ende-Tag
Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + '</TD>'

// +++++ TR erzeugen: Ende-Tag
Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode + '</TR>';
}

// +++++ TD erzeugen +++++
function Anzeige_Kalender_HTMLCode_TD_Erzeugen(
    Inhalt,
    HintergrundFarbe,
    SchriftFarbe,
    SchriftGroesse,
    SchriftArt,
    BorderColor
)

// erweitert Anzeige_Kalender_HTMLCode
// alle Variablen alle global, damit nicht pro Funktionsaufruf die Variablen neu lokal erzeugt werden.
{
// +++++ Inhalt auf FETT setzen
Inhalt='<B>' + Inhalt + '</B>';

// +++++ TD erzeugen
//   Inhalt der TD ist eine Tabelle
Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode
    + '<TD BGCOLOR="' + BorderColor + '">'
    + '<TABLE BORDER=0'
    + ' CELLSPACING=0'
    + ' CELLPADDING=5'
    + ' WIDTH=100%'
    + ' HEIGHT=100%'
    + '>'

Anzeige_Kalender_HTMLCode_TR_Erzeugen(HintergrundFarbe,'30','30','center','middle',"",
    SchriftGroesse,SchriftFarbe,SchriftArt,Inhalt);

Anzeige_Kalender_HTMLCode= Anzeige_Kalender_HTMLCode
    + '</TABLE>'
    + '</TD>';
}

// +++++ Tag (nicht Sonntag) TD erzeugen +++++
function Anzeige_Kalender_Tag_HTMLCode_TD_Erzeugen(Inhalt,Heute)
// erweitert Anzeige_Kalender_HTMLCode
// Heute true, so Tag = heute
// alle Variablen alle global, damit nicht pro Funktionsaufruf die Variablen neu lokal erzeugt werden.
{
// Annahme: Tag ist nicht heute
Anzeige_Kalender_Tag_Kette1=Tag_HintergrundFarbe;
Anzeige_Kalender_Tag_Kette2=Tag_SchriftFarbe;
Anzeige_Kalender_Tag_Kette3=Kalender_Kopf_HintergrundFarbe;

// prüfen auf Tag == heute
if (Heute)
{

```



```

Anzeige_Kalender_Tag_Kette1=Tag_Heute_HintergrundFarbe;
Anzeige_Kalender_Tag_Kette2=Tag_Heute_SchriftFarbe;
Anzeige_Kalender_Tag_Kette3=Tag_SchriftFarbe;
}

Anzeige_Kalender_HTMLCode_TD_Erzeugen(
    Inhalt,
    Anzeige_Kalender_Tag_Kette1,
    Anzeige_Kalender_Tag_Kette2,
    Tag_SchriftGroesse,
    Tag_SchriftArt,
    Anzeige_Kalender_Tag_Kette3
);
}

// +++++ Sonntag TD erzeugen ++++++

function Anzeige_Kalender_SonnTag_HTMLCode_TD_Erzeugen(Inhalt,Sonntag)
// erweitert Anzeige_Kalender_HTMLCode
// Sonntag true, so Tag = Sonntag
// alle Variablen alle global, damit nicht pro Funktionsaufruf die Variablen neu lokal erzeugt werden.
{
    // Annahme: Tag ist nicht Sonntag
    Anzeige_Kalender_Sonntag_Kette1=Tag_HintergrundFarbe;
    Anzeige_Kalender_Sonntag_Kette2=Tag_Sonntag_SchriftFarbe;
    Anzeige_Kalender_Sonntag_Kette3=Kalender_Kopf_HintergrundFarbe;

    // prüfen auf Tag == Sonntag
    if (Sonntag)
    {
        Anzeige_Kalender_Sonntag_Kette1=Tag_Heute_Sonntag_HintergrundFarbe;
        Anzeige_Kalender_Sonntag_Kette2=Tag_Heute_Sonntag_SchriftFarbe;
        Anzeige_Kalender_Sonntag_Kette3=Tag_Sonntag_SchriftFarbe;
    }

    Anzeige_Kalender_HTMLCode_TD_Erzeugen(
        Inhalt,
        Anzeige_Kalender_Sonntag_Kette1,
        Anzeige_Kalender_Sonntag_Kette2,
        Tag_SchriftGroesse,
        Tag_SchriftArt,
        Anzeige_Kalender_Sonntag_Kette3
    );
}

// +++++ Kalender erzeugen und anzeigen ++++++

function Anzeige_Kalender()
// periodischer Aufruf per Timer NUR zur Erkennung des Tages- oder Monats- oder Jahreswechsel, falls
// genau zu diesen Zeitpunkten der Kalender aktiv ist.
// Es spart Ressourcen, wenn Timer nicht benutzt wird !!
// Kalender_Timer_Verwenden auf false für Timer nicht verwenden
// auf true für Timer verwenden
//
// alle Variablen alle global, damit nicht pro Funktionsaufruf die Variablen neu lokal erzeugt werden.
{
    // +++++ aktuelle Zeitangaben holen
    Zeit_AktuelleAngabenHolen(); // füllt
        // Zeit_Jetzt
        // Zeit_Jetzt_Tag_Numerisch
        // Zeit_Jetzt_Monat_Numerisch
        // Zeit_Jetzt_Jahr_Numerisch
        // Zeit_Jetzt_Stunden_Numerisch
        // Zeit_Jetzt_Minuten_Numerisch
        // Zeit_Jetzt_Sekunden_Numerisch
        // Zeit_Vormonat
        // Zeit_VormonatErsterTag

    // +++++ Start- und Endwert der Tage für Erzeugung des Kalenders holen

    // ----- Erster Tag des Vormonats als Startwert
    var Start = Zeit_VormonatErsterTag;
    if(Start > 0)
    {Start--;} // minus 1

```



```

else
{Start = 6;} //

// ----- Endwert für Monatstage ermitteln
// Annahme: Monat hat 31 Tage
var Stop = 31;

// für alle Monate mit 30 Tagen korrigieren
if( (Zeit_Jetzt_Monat_Numerisch==4)
    || (Zeit_Jetzt_Monat_Numerisch==6)
    || (Zeit_Jetzt_Monat_Numerisch==9)
    || (Zeit_Jetzt_Monat_Numerisch==11)
    )
{--Stop;}

// für Februar korrigieren: 29 oder 28 Tage, also Schaltjahr
if(Zeit_Jetzt_Monat_Numerisch==2)
{
    // Februar
    // Annahme: Kein Schaltjahr, also 28 Tage
    Stop=28;

    // Schaltjahr mit 29 Tagen ermitteln
    if ((Zeit_Jetzt_Jahr_Numerisch%4) == 0)
    {Stop++;} // 29

    if ( (Zeit_Jetzt_Jahr_Numerisch%100) == 0)
    {Stop--;} // wieder 28

    if ((Zeit_Jetzt_Jahr_Numerisch%400) ==0 )
    {Stop++;} // 29
}

// +++++ HTML-Code des Kalenders als Tabelle erzeugen

// +++++ äusserste Tabelle also die des Kalenders insgesamt
Anzeige_Kalender_HTMLCode=
    '<TABLE ID="TABLE_Kalender"'
    +      'BORDER=0'
    +      'CELLPADDING=1'
    +      'CELLSPACING=0'
    +      'STYLE="position:absolute;'
    +          'left:' + Kalender_Left + ';'
    +          'top:' + Kalender_Top
    +      '""'
    + '>';

// +++++ 1. TR des Kalenders als Anzeige aktueller Monat und aktuelles Jahr in Fettschrift

// ----- Text der 1. TR ermitteln: aktueller Monat und aktuelles Jahr in Fettschrift
Anzeige_Kalender_Kette='<B>';
Anzeige_Kalender_Kette=MonatsNamenFeld[Zeit_Jetzt_Monat_Numerisch-1]; // Monatsname als String
Anzeige_Kalender_Kette+=' ';
Anzeige_Kalender_Kette+=Zeit_Jetzt_Jahr_Numerisch.toString(); // Jahr zu String
Anzeige_Kalender_Kette+='</B>';

// ----- Abstand zum Nachfolger
Anzeige_Kalender_Kette+='<BR>';
Anzeige_Kalender_Kette+='<BR>';

// ----- und erzeugen
Anzeige_Kalender_HTMLCode_TR_Erzeugen(
    Kalender_Kopf_HintergrundFarbe,"","center','middle','7',
    Kalender_Kopf_SchriftGroesse,
    Kalender_Kopf_SchriftFarbe,
    Kalender_Kopf_SchriftArt,
    Anzeige_Kalender_Kette
);

// +++++ 2. TR Spaltenkopf der Monatstage; pro Spalte ein TD
Anzeige_Kalender_HTMLCode+='<TR>';

// ----- Tage der Woche abklappern: Woche beginnt mit Montag
for(Anzeige_Kalender_TagesZahler=0;

```



```

    Anzeige_Kalender_TagesZahler < TagKurzNamenFeld_Laenge;
    Anzeige_Kalender_TagesZahler++
  )
  {
    // ----- Text als Kurzname des Tages holen
    Anzeige_Kalender_Kette=TagKurzNamenFeld[Anzeige_Kalender_TagesZahler];

    // ----- und Text erzeugen
    Anzeige_Kalender_HTMLCode_TD_Erzeugen(
      Anzeige_Kalender_Kette,
      Kalender_Kopf_HintergrundFarbe,
      Kalender_SpaltenKopf_SchriftFarbe,
      Tag_SchriftGroesse,
      Kalender_Kopf_SchriftArt,
      Kalender_Kopf_HintergrundFarbe
    );
  }

  Anzeige_Kalender_HTMLCode+="|"; // Ende der Wocheneinteilung in 7 Tage (Spaltenköpfe)

  // +++++ ab 3. TR erfolgt die Wochenauflistung
  //   pro Woche 1 Zeile
  //   pro Zeile 1 TR
  //   ab 3. TR im gesamten Kalender

  // ----- init des Tageszähler des Monats
  Anzeige_Kalender_TagesZahler = 1;

  // ----- Wochen abklappen also Zeilen (TR)
  for(Anzeige_Kalender_Zahler_TR=0;
    Anzeige_Kalender_Zahler_TR < 6;
    Anzeige_Kalender_Zahler_TR++
  )
  {
    // - - - aktuelle Woche (Zeile) erzeugen
    Anzeige_Kalender_HTMLCode+="|";

    // - - - pro Spalte 1 TD erzeugen: Nur die ersten 6 Tage der Woche, OHNE Sonntag !
    for(Anzeige_Kalender_Zahler_TD=0;
      Anzeige_Kalender_Zahler_TD < 6; // ohne Sonntag !
      Anzeige_Kalender_Zahler_TD++
    )
    {
      // - - - prüfen ob 1. TR-Zeile noch nicht abgeklappert wurde
      if( (Anzeige_Kalender_Zahler_TR == 0)
        && (Anzeige_Kalender_Zahler_TD < Start)
      )
      {
        // 1. TR und nur 1. TD
        Anzeige_Kalender_Tag_HTMLCode_TD_Erzeugen('&#160;','false');
      }
      else
      {
        // 1. TR ab 2.TD oder ab 2.TR

        // - - - prüfen ob alle Tage bereits abgeklappert wurden
        if (Anzeige_Kalender_TagesZahler > Stop)
        {
          // alle Tage sind abgeklappert
          Anzeige_Kalender_Tag_HTMLCode_TD_Erzeugen('&#160;','false');
        }
        else
        {
          // 1. TR ab 2.TD oder ab 2.TR
          // es sind noch Tage abzuklappen

          // - - - prüfen ob aktueller Tag der Tag von JETZT ist, also Heute ist
          if(Anzeige_Kalender_TagesZahler==Zeit_Jetzt_Tag_Numerisch)
          {
            // 1. TR ab 2.TD oder ab 2.TR
            // es sind noch Tage abzuklappen
            // aktueller Tag ist heute

            Anzeige_Kalender_Tag_HTMLCode_TD_Erzeugen(Anzeige_Kalender_TagesZahler,true);

|  |

|  |

```



```

    }
    else
    {
        // 1. TR ab 2.TD oder ab 2.TR
        // es sind noch Tage abzuklappen
        // aktueller Tag ist nicht heute
        Anzeige_Kalender_Tag_HTMLCode_TD_Erzeugen(Anzeige_Kalender_TagesZahler,false);
    }

    // - - - Tageszähler erhöhen für nächsten Tag als neuen aktuellen
    Anzeige_Kalender_TagesZahler++;
    }
}

// - - - ALLE TD der ersten 6 Tage der aktuellen Woche (Zeile) wurden erzeugt
//      offen ist der 7. Tag, also der Sonntag (falls vorhanden)

//      prüfen ob alle Tage abgeklappert wurden, also ob es keinen Sonntag gibt
if(Anzeige_Kalender_TagesZahler > Stop)
{
    // keine Tage mehr abzuklappen, kein Sonntag mehr da
    Anzeige_Kalender_SonnTag_HTMLCode_TD_Erzeugen('&#160;';false);
}
else
{
    // - - - Es sind noch Tage abzuklappen, also 7. Tag = Sonntag vorhanden

    // - - - prüfen ob der aktuelle Tag der von heute ist,
    //      wenn ja, dann ist heute Sonntag
    if(Anzeige_Kalender_TagesZahler==Zeit_Jetzt_Tag_Numerisch)
    {
        // heute ist Sonntag, also den Tag mit anderem Hintergrund erzeugen
        // als alle anderen Sonntage
        Anzeige_Kalender_SonnTag_HTMLCode_TD_Erzeugen(Anzeige_Kalender_TagesZahler,true);
    }
    else
    {
        // heute ist nicht Sonntag, also den Tag dem normalen Hintergrund für Sonntage erzeugen
        Anzeige_Kalender_SonnTag_HTMLCode_TD_Erzeugen(Anzeige_Kalender_TagesZahler,false);
    }

    // - - - Tageszähler erhöhen für nächsten Tag als neuen aktuellen
    Anzeige_Kalender_TagesZahler++;
}

// - - - aktuelle Woche also TD beenden
Anzeige_Kalender_HTMLCode+='</TR>';
}

// +++++ Ende des Kalenders
Anzeige_Kalender_HTMLCode+='</TABLE>';

// +++++ und Kalenderdaten neu rendern
document.all.ID_Kalender.innerHTML = Anzeige_Kalender_HTMLCode;

// +++++ prüfen ob der Timer zur Erkennung der Tages-, Monats- und Jahreswechsel
//      verwendet werden soll
if (Kalender_Timer_Verwenden)
{window.setTimeout('Anzeige_Kalender()',5000);}
}

// -->
</SCRIPT>
</HEAD>

<BODY BGCOLOR=#3A6EA5>
<!-- BODY mit identischer Hintergrundfarbe wie laut Kalender_Kopf_HintergrundFarbe -->

<DIV ID="ID_Kalender"></DIV>
<!-- Der DIV wird nach seiner Erzeugung erst gefüllt: per .innerHTML -->

<SCRIPT LANGUAGE="JScript">
<!--

```



```
Anzeige_Kalender();
// -->
</SCRIPT>
</BODY>
</HTML>
```

4.3.2.2.4.3.42.6. Dynamische Veränderung einer Tabelle in JScript

Die Manipulation der Tabelle ist erst möglich, wenn das Dokument komplett geladen und das Event onload ausgelöst wurden.

4.3.2.2.4.3.42.6.1. Datenveränderung per JScript

auf Basis einer in Script kodierten Struktur und zur Laufzeit des Dokumentes, das die Tabelle besitzt nur in der jeweiligen Zelle einer Zeile (siehe oben)

Nur bei einer Zelle sind die Eigenschaften `.innerHTML` bzw. `.innerText` schreibbar.

4.3.2.2.4.3.42.6.2. Layoutveränderung per Style

auf Basis einer in Script kodierten Struktur und zur Laufzeit des Dokumentes, das die Tabelle besitzt werden am schnellsten und automatisch gerendert

Tabellen-Refresh nicht nötig

alle Tabellenelement wie die Tabelle selbst besitzen das STYLE-Attribut
siehe sonst oben

4.3.2.2.4.3.42.6.3. Strukturveränderung per JScript

auf Basis einer in Script kodierten Struktur und zur Laufzeit des Dokumentes, das die Tabelle besitzt
siehe oben

4.3.2.2.4.3.42.7. Eigenschaften der Tabelle

<code>.accessKey</code>	Tastaturzugriff auf ein Objekt per Alt + Taste
<code>.align</code>	Ausrichtung
<code>ATOMICSELECTION</code>	Selektierbarkeit des Objektes einstellen
<code>.background</code>	Hintergrundbild eines Objektes
<code>.begin</code>	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft <code>.timeAction</code> siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code>
<code>.bgColor</code>	deprecated und durch STYLE-Attribut zu ersetzen
<code>.border</code>	Rahmendicke in Pixel
<code>.borderColor</code>	Borderfarbe (Rahmenfarbe)
<code>.borderColorDark</code>	deprecated und durch Eigenschaft <code>.borderColor</code> zu ersetzen dunkle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft <code>.border</code> (nicht Style-Eigenschaft <code>border</code>)
<code>.borderColorLight</code>	deprecated und durch Eigenschaft <code>.borderColor</code> zu ersetzen helle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft <code>.border</code> (nicht Style-Eigenschaft <code>boder</code>)
<code>.canHaveChildren</code>	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
<code>.canHaveHTML</code>	prüfen ob Objekt HTML-Tags enthalten darf
<code>.caption</code>	Zeiger auf das Objekt <code>table.caption</code> es darf nur 1 CAPTION zur Tabelle existieren
<code>.cellPadding</code>	Abstand zwischen Zellrahmen und dem Inhalt der Zelle
<code>.cellSpacing</code>	Abstand zwischen den Zellen

Beispiel:

```
<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
  <TR>
    <TD>Zelle 1</TD>
    <TD>Zelle 2</TD>
  </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.cellSpacing=20">20 </BUTTON>
<BUTTON onclick="ID_Tabelle.cellSpacing=5">5 </BUTTON>
```

<code>.className</code>	Klassenreferenz, Klassenname
<code>.clientHeight</code>	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
<code>.clientLeft</code>	Abstand in Pixel zum linken Rand des Fensters
<code>.clientTop</code>	Abstand in Pixel zum oberen Rand des Fensters
<code>.clientWidth</code>	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
<code>.cols</code>	Anzahl der Spalten in der Tabelle wenn belegt so wird Tabelle schneller gerendert

Beispiel:

```
<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
  <TR>
    <TD>Zelle 1</TD>
    <TD>Zelle 2</TD>
  </TR>
</TABLE>
<BUTTON onclick="alert(ID_Tabelle.cols);">Anzahl</BUTTON>
<BUTTON onclick="alert(ID_Tabelle.dataPageSize);">Anzahl</BUTTON>
```

`.dataPageSize`

Anzahl der sichtbaren Datensätze auf einer Tabellenseite
Anzahl der Sätze pro Dataset-Anzeige

Beispiel:

Datensätze liegen in der Datei `adress.txt`
Datei `adress.txt` liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument



hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815 Willmann;Theo;1234 Xantippe;Isa;5678 Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) {Kette = "+";}

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition ==1)
        {alert("Erster Datensatz erreicht!");}
    }

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
```



```

        DATASRC=#ID_Datenbank
        DATAPAGESIZE=1
    >
        <THEAD>
            <TR>
                <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
                <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
                <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
            </TR>
        </THEAD>
        <TBODY>
            <TR>
                <TD><DIV DATAFLD="vorname"></DIV></TD>
                <TD><DIV DATAFLD="name"></DIV></TD>
                <TD><DIV DATAFLD="telefon"></DIV></TD>
            </TR>
        </TBODY>
    </TABLE>
    <BR>
    <INPUT          TYPE="button"
                    VALUE="Zurueck"
                    onclick=rueckwaerts(1)"
    >
    <!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->

    <INPUT          TYPE="button"
                    VALUE="Vorwaerts"
                    onclick=vorwaerts(1)"
    >
    <!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->
</BODY>
</HTML>

.dataSrc      Datenquelle als Anker festlegen
.dir          Umflussrichtung
.disabled     Interaktionsfähigkeit
              nur wenn sichtbar so User-Interaktion möglich
.end          Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild   Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.frame        Art des Rahmens um eine Tabelle
              "void"      Standard, kein Rahmen
              "above"     Rahmen oberhalb
              "below"     Rahmen unterhalb
              "border"    Rahmen auf allen Seiten
              "box"       Rahmen auf allen Seiten
              "hsides"    Rahmen oben und unten
              "lhs"       Rahmen links
              "rhs"       Rahmen rechts
              "vsides"    Rahmen links und rechts

.hasMedia     Objekt ist HTML-Media-Objekt
.height       Höhe des Objektes in Pixel
.hideFocus    Focussierbarkeit
.id           Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
              ID-Attribut in HTML: Wert ist String      alphanumerisch
                                                         muss mit Buchstaben beginnen
                                                         Unterstrich _ verwendbar
                                                         kann in " " bzw. ' ' kodiert werden, muss aber nicht

Hinweis:      Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt
              und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und
              betroffenes Objekt die Eigenschaft .uniqueID kennen).

Zeiger aus ID bilden  var Zeiger = eval(object.id);
Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.

.innerHTML     Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
              ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. <B>
              dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
                  also nach dem Laden des Objektes
                  aber wirksam erst mit parsen des HTML-Endetag

Plain-Text
HTML-Elemente je nach Objekt möglich
Script: muss mit DEFER-Attribut kodiert sein
schreiben: wenn Bereich nicht leer, so komplett überschreiben
              wenn Bereich leer so einfügen
nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD,
              TITLE, TR.
lesen erfolgt anhand der Angaben im Quelltext und laut Lage des Objektes
.innerText     Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
              ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. <B>

```



	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isContentEditable	Mehrzeiligkeit des Objekthinhalte
.isDisabled	Erzeugbarkeit eines Textbereiches
.isMultiLine	Sprache für Anzeige von Sonderzeichen etc.
.isTextEdit	Sprache für Script festlegen
.lang	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.language	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.lastChild	String als Name des Kindes (Knoten, Node, Element)
.nextSibling	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeName	Knotentyp laut attributes Collection
.nodeType	1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.rules	Art der sichtbaren inneren Rahmen einer Tabelle Art der sichtbaren Rahmen zwischen den Tabellenelementen Art des sichtbaren Rahmens um Tabelle "all" Rahmen um alle Zeilen und Spalten "cols" Trennlinie zwischen allen Spalten "groups" horizontale Trennlinie zwischen allen THEAD, TBODY's und TFOOT vertikale Trennlinie zwischen allen COLGROUP "none" keine Rahmen und Trennlinien zwischen Tabellenelementen "rows" Trennlinie zwischen allen Zeilen "" keine Rahmen generell (auch nicht um Tabelle)

Beispiel:

```
<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
<THEAD>
```



```

        <TR>
            <TD>Kopf Zelle 1</TD>
            <TD>Kopf Zelle 2</TD>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD>Body Zeile 1 Zelle 1</TD>
            <TD>Body Zeile 1 Zelle 2</TD>
        </TR>
        <TR>
            <TD>Body Zeile 2 Zelle 1</TD>
            <TD>Body Zeile 2 Zelle 2</TD>
        </TR>
        <TR>
            <TD>Body Zeile 3 Zelle 1</TD>
            <TD>Body Zeile 3 Zelle 2</TD>
        </TR>
    </TBODY>
</TFOOT>
<TR>
    <TD>Fuss Zelle 1</TD>
    <TD>Fuss Zelle 2</TD>
</TR>
</TFOOT>
</TABLE>
<BUTTON onclick=ID_Tabelle.rules="";>keine Rahmen</BUTTON>
<BUTTON onclick=ID_Tabelle.rules="none";>none</BUTTON>
<BUTTON onclick=ID_Tabelle.rules="all";>all</BUTTON>
<BUTTON onclick=ID_Tabelle.rules="cols";>cols</BUTTON>
<BUTTON onclick=ID_Tabelle.rules="groups";>groups</BUTTON>
<BUTTON onclick=ID_Tabelle.rules="rows";>rows</BUTTON>

```

.scopeName Namensraum laut XMLNS-Attribut

.scrollHeight Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes

.scrollLeft Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes

.scrollTop immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes

.scrollWidth immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes

.sourceIndex Index des Objektes in der Collection document.all

STYLE direkt im HTML-Element kodierter Style (Inline-Style)

.summary Hinweis: für Scripting ist das Style-Objekt zu nutzen Kommentar in einer Tabelle, der nicht gerendert wird

Beispiel:

```

<TABLE ID="ID_Tabelle" BORDER CELSPACING=10 SUMMARY="Kommentar">
    <TR>
        <TD>Zelle 1</TD>
        <TD>Zelle 2</TD>
    </TR>
</TABLE>

```

.syncMaster Synchronisierung der Animation des im Container liegenden Elementes auf Timeline

.systemBitrate wird hier nicht erklärt

.systemCaptions wird hier nicht erklärt

.systemLanguage Sprache festlegen für das Objekt

.systemOverdubOrSubtitle wird hier nicht erklärt

.tabIndex Index des Elementes in der Tab-Tasten-Folge

.tagName Tag-Bezeichner des Objektes

.tagUrn Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut

.tfoot Zeiger auf das Objekt table.tFoot

.thead Zeiger auf das Objekt table.tHead

.timeContainer Typ der Timeline des Objektes

.title siehe Objekt currTimeState und Behavior .style.time2

.uniqueID Tooltip-Text bei Mouse over über Objekt

.width durch den Browser automatisch-generiertes ID des Objektes

UNSELECTABLE Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

.width Selektionsfähigkeit eines Objektes

.width Breite des Objektes in Pixel

4.3.2.2.4.3.42.8. Methoden der Tabelle

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen



	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbare
.click()	DOM wird geändert simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.createCaption()	leeres CAPTION in einer Tabelle erzeugen und einbinden es darf nur 1 CAPTION zur Tabelle existieren
.createTFoot()	leeres TFOOT in einer Tabelle erzeugen und einbinden es darf nur 1 TFOOT zur Tabelle existieren
.createTHead()	leeres THEAD in einer Tabelle erzeugen und einbinden es darf nur 1 THEAD zur Tabelle existieren
.deleteCaption()	CAPTION löschen aus Tabelle es darf nur 1 CAPTION zur Tabelle existieren
.deleteRow()	Zeile löschen aus Tabelle siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
.deleteTFoot()	TFOOT der Tabelle löschen es darf nur 1 TFOOT zur Tabelle existieren siehe Objekt table siehe Objekt table.tBody
.deleteTHead()	THEAD der Tabelle löschen es darf nur 1 THEAD zur Tabelle existieren siehe Objekt table siehe Objekt table.tBody
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.firstPage()	erste Seite des Datasets in Tabelle anzeigen Eigenschaft .dataPageSize muss belegt worden sein

Beispiel:



Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815
	Willmann;Theo;1234
	Xantippe;Isa;5678
	Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) { Kette = "+"; }

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            { ID_Datenbank.recordset.MoveNext(); }
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        { alert("Letzter Datensatz erreicht!"); }
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            { ID_Datenbank.recordset.MovePrevious(); }
        }

        if (ID_Datenbank.recordset.AbsolutePosition == 1)
        { alert("Erster Datensatz erreicht!"); }
    }

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>
```



```

<TABLE ID="ID_Tabelle"
  DATASRC=#ID_Datenbank
  DATAPAGESIZE=1
>
  <THEAD>
    <TR>
      <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
      <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
      <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
    </TR>
  </THEAD>
  <TBODY>
    <TR>
      <TD><DIV DATAFLD="vorname"></DIV></TD>
      <TD><DIV DATAFLD="name"></DIV></TD>
      <TD><DIV DATAFLD="telefon"></DIV></TD>
    </TR>
  </TBODY>
</TABLE>
<BR>
<INPUT          TYPE="button"
                VALUE="Zurueck"
                onclick=rueckwaerts(1)"
>
<INPUT          TYPE="button"
                VALUE="Vorwaerts"
                onclick=vorwaerts(1)"
>
</BODY>
</HTML>

```

<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->

<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->

.focus() Focus setzen und Focus-Event auslösen
nur nach dem kompletten Laden des Dokumentes
vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

.getAdjacentText() Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
Text kann HTML-Tags enthalten, muss aber nicht
DOM nicht geändert

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
DOM nicht geändert

.getAttributeNode() Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft.
Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht
Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt
Wert des Attributes wird somit über die Referenz laut DOM erreichbar
DOM nicht geändert

.getBoundingClientRect() Referenz auf TextRectangle-Objekt im Element holen

.getClientRects() Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster
Feld mit Index als Integer ab 0
pro Eintrag ein Rectangle

.getElementsByTagName() Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen
Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.
Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden
(beachte dabei document.all Collection)
Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !
Für Verwaltung per ID (analog zum ID-Attribut):
siehe Methode getElementById()
Für Verwaltung per NAME (analog zum NAME-Attribut):
siehe Methode .getElementsByName()

.getExpression() DOM nicht geändert
Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern
Style-Eigenschaft ist per Methoden
expression() oder setExpression()
zu definieren
DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout
(nach dem eventuellen expliziten Dokument-Refresh)

.hasChildNodes() prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten
(Textelemente) in einem Objekt
DOM nicht geändert

.insertAdjacentElement() Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann
wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM
nur nach dem kompletten Laden des Dokumentes möglich
DOM wird geändert

.insertBefore() Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern
einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein



Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten
Sichtbarkeit erst wenn Ende-Tag geparkt wurde
DOM wird geändert

.insertRow()

Zeile in die Tabelle einfügen
siehe Objekt table
siehe Objekt table.tBody
siehe Objekt table.tHead
siehe Objekt table.tFoot

.lastPage()

letzte Seite des Datasets in Tabelle anzeigen
Eigenschaft .dataPageSize muss belegt worden sein

Beispiel:

Datensätze liegen in der Datei adress.txt
Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815
	Willmann;Theo;1234
	Xantippe;Isa;5678
	Arktin;Richard;1055

<HTML>

<HEAD>

<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">

<!--

```
var SortierRichtung = true;
```

```
function Sortieren(FeldBezeichner)
```

```
{
```

```
    // Sortierrichtung festlegen
```

```
    var Kette = "-"; // Annahme
```

```
    if (SortierRichtung) { Kette = "+"; }
```

```
    // nächste Sortierung umgekehrt
```

```
    SortierRichtung = !SortierRichtung;
```

```
    // sortieren
```

```
    ID_Datenbank.Sort= Kette + FeldBezeichner;
```

```
    // und das Ergebnis anzeigen
```

```
    ID_Datenbank.Reset();
```

```
}
```

```
function vorwaerts(AnzahlSaetze)
```

```
{
```

```
    // nächste Seite anzeigen
```

```
    document.all.ID_Tabelle.nextPage();
```

```
    // und Satzzeiger korrigieren
```

```
    for (var i = 0; i < AnzahlSaetze; i++)
```

```
    {
```

```
        if (ID_Datenbank.recordset.AbsolutePosition !=
```

```
            ID_Datenbank.recordset.RecordCount)
```

```
        { ID_Datenbank.recordset.MoveNext(); }
```

```
    }
```

```
    if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
```

```
    { alert("Letzter Datensatz erreicht!"); }
```

```
}
```

```
function rueckwaerts(AnzahlSaetze)
```

```
{
```

```
    // vorhergehende Seite anzeigen
```

```
    document.all.ID_Tabelle.previousPage();
```

```
    // und Satzzeiger korrigieren
```

```
    for (var i = 0; i < AnzahlSaetze; i++)
```

```
    {
```

```
        if (ID_Datenbank.recordset.AbsolutePosition > 1)
```

```
        { ID_Datenbank.recordset.MovePrevious(); }
```

```
    }
```

```
    if (ID_Datenbank.recordset.AbsolutePosition == 1)
```

```
    { alert("Erster Datensatz erreicht!"); }
```

```
}
```



```
// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
>
    <THEAD>
        <TR>
            <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
            <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
            <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD><DIV DATAFLD="vorname"></DIV></TD>
            <TD><DIV DATAFLD="name"></DIV></TD>
            <TD><DIV DATAFLD="telefon"></DIV></TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->
</BODY>
</HTML>
```

.mergeAttributes() alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
 Attribute sind: HTML
 Events
 Styles
 ab IE 5.01 auch ID, NAME

Achtung: Diese Methode ist mir Vorsicht zu geniessen !!

.moveRow() DOM wird geändert
 2 Zeilen in der Tabelle austauschen
 siehe Objekt table
 siehe Objekt table.tBody
 siehe Objekt table.tHead
 siehe Objekt table.tFoot

Beispiel:

```
<TABLE ID="ID_Tabelle" BORDER CELLSPACING=10>
    <TR>
        <TD>Body Zeile 1 Zelle 1</TD>
        <TD>Body Zeile 1 Zelle 2</TD>
    </TR>
    <TR>
        <TD>Body Zeile 2 Zelle 1</TD>
        <TD>Body Zeile 2 Zelle 2</TD>
    </TR>
    <TR>
        <TD>Body Zeile 3 Zelle 1</TD>
        <TD>Body Zeile 3 Zelle 2</TD>
    </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.moveRow(0,1);">verschieben Zeile 0 auf Zeile 1</BUTTON>
```

.nextPage() nächste Seite des Datasets in Tabelle anzeigen
 Eigenschaft .dataPageSize muss belegt worden sein

Beispiel:



Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815
	Willmann;Theo;1234
	Xantippe;Isa;5678
	Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) {Kette = "+";}

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition ==1)
        {alert("Erster Datensatz erreicht!");}
    }

    // -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>
```




```

<TABLE ID="ID_Tabelle"
  DATASRC=#ID_Datenbank
  DATAPAGESIZE=1
>
  <THEAD>
    <TR>
      <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
      <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
      <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
    </TR>
  </THEAD>
  <TBODY>
    <TR>
      <TD><DIV DATAFLD="vorname"></DIV></TD>
      <TD><DIV DATAFLD="name"></DIV></TD>
      <TD><DIV DATAFLD="telefon"></DIV></TD>
    </TR>
  </TBODY>
</TABLE>
<BR>
<INPUT      TYPE="button"
              VALUE="Zurueck"
              onclick=rueckwaerts(1)"
>
<INPUT      TYPE="button"
              VALUE="Vorwaerts"
              onclick=vorwaerts(1)"
>
</BODY>
</HTML>

```

.normalize() Normalisierung des DOM zur Erreichung einer konsistenten Struktur
 Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
 .previousPage() vorhergehende Seite des Datasets in Tabelle anzeigen
 Eigenschaft .dataPageSize muss belegt worden sein

Beispiel:

Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815 Willmann;Theo;1234 Xantippe;Isa;5678 Arktin;Richard;1055

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
  var SortierRichtung = true;

  function Sortieren(FeldBezeichner)
  {
    // Sortierrichtung festlegen
    var Kette = "-"; // Annahme
    if (SortierRichtung) { Kette = "+"; }

    // nächste Sortierung umgekehrt
    SortierRichtung = !SortierRichtung;

    // sortieren
    ID_Datenbank.Sort= Kette + FeldBezeichner;

    // und das Ergebnis anzeigen
    ID_Datenbank.Reset();
  }

  function vorwaerts(AnzahlSaetze)
  {
    // nächste Seite anzeigen
    document.all.ID_Tabelle.nextPage();

    // und Satzzeiger korrigieren

```



```

        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == 1)
        {alert("Erster Datensatz erreicht!");}
    }

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC="#ID_Datenbank"
    DATAPAGESIZE=1
>
    <THEAD>
        <TR>
            <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
            <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
            <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD><DIV DATAFLD="vorname"></DIV></TD>
            <TD><DIV DATAFLD="name"></DIV></TD>
            <TD><DIV DATAFLD="telefon"></DIV></TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->
</BODY>
</HTML>

.refresh()
    Anzeige der Tabelle neu erzeugen
    Änderungen an der Tabelle sichtbar machen z.B. wenn Tabelle in Anzahl Zeilen bzw. Spalten
    manipuliert wurde
    siehe Objekt table
.releaseCapture()
    Maus-Überwachung ausschalten für ein Objekt

```



	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute ! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setCapture()	DOM wird geändert Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.9. *table.caption Objekt des Internet Explorer*

Tabellenüberschrift

Es kann nur 1 CAPTION für eine Tabelle kodiert werden

Beispiel:

```
<TABLE>
  <CAPTION VALIGN=BOTTOM>
    Überschrift
  </CAPTION>
  ...
</TABLE>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.background	Hintergrundbild eines Objektes
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction



	siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	ID-Attribut in HTML: Wert ist String
	alphanumerisch
	muss mit Buchstaben beginnen
	Unterstrich _ verwendbar
	kann in " " bzw. ' ' kodiert werden, muss aber nicht
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt
	und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und
	betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
	Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetag
	Plain-Text
	HTML-Elemente je nach Objekt möglich
	Script: muss mit DEFER-Attribut kodiert sein
	schreiben: wenn Bereich nicht leer, so komplett überschreiben
	wenn Bereich leer so einfügen
	nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD,
	TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetag
	nur Plain-Text also keine HTML-Elemente und kein Script
	schreiben: wenn Bereich nicht leer, so komplett überschreiben
	wenn Bereich leer so einfügen
	nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
	1 für Element-Knoten.
	3 für Textknoten.
	oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern
	für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
	des Elternobjektes (.offsetParent)



.onOffBehavior	deprecated ab IE 5.x
.outerHTML	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vAlign	Lage der CAPTION einer Tabelle es darf nur 1 CAPTION zur Tabelle existieren nach Änderung ist ein Tabellen-Refresh per Methode .refresh() notwendig "top" Überschrift am Kopf der Tabelle Standard "bottom" Überschrift am Fuss der Tabelle

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler



	Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM- Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert



<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> .
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
<code>.setExpression()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-



Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 .swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.42.10. table.col Objekt des Internet Explorer

Spalte einer Tabelle
 kann innerhalb von COLGROUP kodiert sein:

COL **erbt keine** gleichnamige Eigenschaften von COLGROUP
 überschreibt gleichnamige Eigenschaften von COLGROUP

Beispiel:

```
<TABLE BORDER="2">
  <COL SPAN="2" STYLE="color:red">
  <COL STYLE="color:blue">
```

```
.....
</TABLE>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id); Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
	Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)



.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.span	Anzahl der Spalten innerhalb einer Spaltengruppe einer Tabelle

Beispiel:

```

<TABLE BORDER="2">
  <COLGROUP SPAN="3" STYLE="color:green;background:black">
    <COL SPAN="2" STYLE="color:red">
  </COLGROUP>
  ....
</TABLE>

```

STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf
.width	Breite des Objektes in Pixel Integer in Pixels für absolute Breite, >=0 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel String Ziffernfolge eines Integer mit nachfolgendem % für Breite als Anteil der Breite des Elternobjektes z.B. 10%

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
----------------	---



	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernenbar DOM wird geändert
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelposition erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName() DOM nicht geändert
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt



.insertAdjacentElement()	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
.insertBefore()	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.11. table.colGroup Objekt des Internet Explorer

Objekt zur Gruppierung von Spalten einer Tabelle

COL kann innerhalb von COLGROUP kodiert sein:

COL **erbt keine** gleichnamige Eigenschaften von COLGROUP
überschreibt gleichnamige Eigenschaften von COLGROUP

Bsp: Es sollte das SPAN-Attribut innerhalb von COL kodiert werden, wenn SPAN in COLGROUP mit einem anderen Wert hat als dem Standardwert des SPAN-Attributes von COL kodiert wurde.

Beispiel 1:

```
<TABLE BORDER="2" RULES="groups">
```



```

<COLGROUP SPAN="2" STYLE="color:red">
</COLGROUP>
<COLGROUP STYLE="color:blue">
</COLGROUP>

```

```

....
</TABLE>

```

Beispiel 2:

```

<TABLE BORDER="2">
<COLGROUP SPAN="3" STYLE="color:green;background:black">
<COL SPAN="2" STYLE="color:red">
</COLGROUP>
....
</TABLE>

```

Eigenschaften:

.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.dir	Umflussrichtung
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
	Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
	Plain-Text
	HTML-Elemente je nach Objekt möglich
	Script: muss mit DEFER-Attribut kodiert sein
	schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen
	nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)



.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.span	Anzahl der Spalten innerhalb einer Spaltengruppe einer Tabelle
Beispiel:	
<pre> <TABLE BORDER="2"> <COLGROUP SPAN="3" STYLE="color:green;background:black"> <COL SPAN="2" STYLE="color:red"> </COLGROUP> </TABLE> </pre>	
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf
.width	Breite des Objektes in Pixel Integer in Pixels für absolute Breite, >=0 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel String Ziffernfolge eines Integer mit nachfolgendem % für Breite als Anteil der Breite des Elternobjektes z.B. 10%
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern



	DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode <code>.createElement()</code> erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft <code>.innerHTML</code> gelöscht !
<code>.attachEvent()</code>	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode <code>.detachEvent()</code> Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
<code>.clearAttributes()</code>	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbare
<code>.cloneNode()</code>	DOM wird geändert Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.contains()</code>	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
<code>.detachEvent()</code>	DOM nicht geändert Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_bezeichner</code> zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
<code>.fireEvent()</code>	ein Event auslösen
<code>.getAdjacentText()</code>	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
<code>.getAttribute()</code>	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
<code>.getAttributeNode()</code>	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
<code>.getBoundingClientRect()</code>	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
<code>.getClientRects()</code>	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
<code>.getElementsByName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei <code>document.all</code> Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.getExpression()</code>	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden <code>expression()</code> oder <code>setExpression()</code> zu definieren
<code>.hasChildNodes()</code>	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
<code>.insertAdjacentElement()</code>	DOM nicht geändert Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich
<code>.insertBefore()</code>	DOM wird geändert Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde



.mergeAttributes()	DOM wird geändert alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !!
.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.removeChild()	DOM wird geändert Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
.removeExpression()	DOM wird geändert Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
.removeNode()	DOM wird nicht geändert Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.replaceChild()	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
.replaceNode()	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
.scrollIntoView()	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.setExpression()	DOM wird geändert Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient Ausdruck nur als Script kodierbar
.swapNode()	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.12. *table.rows Collection des Internet Explorer*

referenziert alle Zeilen der Tabelle, egal wo diese liegen (THEAD etc.), in der Reihenfolge der Zeilen in der Tabelle
siehe auch Collection `table.tBody.rows`

`table.tFoot.rows`
`table.tHead.rows`

siehe auch Objekt `table.tr` und dort die Eigenschaften `.rowIndex` und `.sectionRowIndex`

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_tabelle.rows  
[ var ZeigerAufFeldElement = ] zeiger_auf_tabelle.rows[Index]
```

Index Integer, ab 0
muss in [] kodiert werden

`zeiger_auf_tabelle` laut ID-Attribut

Beispiel:

```
var CollectionRows = zeiger_auf_tabelle.rows;  
var AnzahlElementeInDerCollectionRows = CollectionRows.length;
```



```
for (var i = 0; i < AnzahlElementeInDer CollectionRows; i++)
{ CollectionRows [i].style.fontWeight = "bold"; }
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.42.13. table.rows.cells Collection des Internet Explorer

referenziert alle Zellen genau einer Zeile laut Reihenfolge der Zellen in der Zeile

Jede Zeile ist eine Element in der Collection table.rows

hat eine eigene Collection table.rows.cells.

siehe auch Collection table.tBody.rows.cells

table.tFoot.rows.cells

table.tHead.rows.cells

siehe auch Objekt table.tr und dort die Eigenschaften .rowIndex und .sectionRowIndex

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_tabelle.rows[Index1].cells
[ var ZeigerAufFeldElement = ] zeiger_auf_tabelle.rows[Index1].cells[Index2]
```

Index1 Integer, ab 0
muss in [] kodiert werden

Index2 Integer, ab 0
muss in [] kodiert werden

zeiger_auf_tabelle laut ID-Attribut

```
[ var ZeigerAufFeld = ] zeiger_auf_zeile.cells
[ var ZeigerAufFeldElement = ] zeiger_auf_zeile.cells[Index]
```

Index Integer, ab 0
muss in [] kodiert werden

zeiger_auf_zeile laut ID-Attribut bzw. laut Erzeugung

Beispiel:

```
var CollectionRows = zeiger_auf_tabelle.rows;
var AnzahlElementeInDer CollectionRows = CollectionRows.length;

for (var i = 0; i < AnzahlElementeInDer CollectionRows; i++)
{
    var AktuelleZeile = CollectionRows [i];
    var AnzahlZellenInAktuelleZeile = AktuelleZeile.length;

    for (var j = 0; j < AnzahlZellenInAktuelleZeile; j++)
    {
        var AktuelleZelleDerZeile = AktuelleZeile.cells[j];

        AktuelleZelleDerZeile.style.width = "20";
    }
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.42.14. table.tBody Objekt des Internet Explorer

Tabellenkörper TBODY

kann folgende innere Elemente als Kinder haben: TD, TH, TR

TBODY muss nicht kodiert werden, wenn keine Fehlzuordnung zu THEAD und kein TFOOT möglich ist



Beispiel:

```
<TABLE>
  <THEAD>
    <TR>
      <TD>Head</TD>
    </TR>
  </THEAD>
  <TBODY>
    <TR>
      <TD>Body</TD>
    </TR>
  </TBODY>
</TABLE>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.background	Hintergrundbild eines Objektes
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.dir	Umflussrichtung
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id); Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektkinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)



	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes



"middle" zentriert, Standard
 "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des **benachbarten** Objektes
 "bottom" am Fuss
 "top" am Kopf

Methoden:

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen
 Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
 ab IE 5.x bis unter IE 5.5

.appendChild() Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
 DOM wird geändert
 Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
 Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde

.applyElement() Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
 DOM wird geändert
 Element kann selbst Kinder haben
 Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde
 Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in **Zufallsfolge**, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)

.blur() Element den Focus wegnehmen und Event onblur auslösen
 Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
 vor IE 5.0 TABINDEX-Attribut muss kodiert sein
 ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein

.clearAttributes() alle HTML-Attribute eines Objektes entfernen
 außer ID, STYLE und per Script definierte Attribute
 Script-erzeugte Attribute nicht entfernen
 DOM wird geändert

.click() simuliert einen Klick auf das Element und löst onclick-Event aus
 manipuliert nicht den Focus

.cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
 DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)

.componentFromPoint() Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
 auch für CSS-Layout
 onmouseover-Event hat **nicht die Pixelgenauigkeit** wie die Angaben der Methode .componentFromPoint()
 also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben
 Overbereich der Maus ist mehr als 1 Pixel gross
 beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.

.contains() prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
 DOM nicht geändert

.deleteRow() Zeile löschen aus Tabelle
 siehe Objekt table
 siehe Objekt table.tBody
 siehe Objekt table.tHead
 siehe Objekt table.tFoot

.deleteTFoot() TFOOT der Tabelle löschen
 es darf nur 1 TFOOT zur Tabelle existieren
 siehe Objekt table
 siehe Objekt table.tBody

.deleteTHead() THEAD der Tabelle löschen
 es darf nur 1 THEAD zur Tabelle existieren
 siehe Objekt table
 siehe Objekt table.tBody

.detachEvent() Abschalten des Registrieren eines Events durch Eventhandler
 wobei Registrierung mit Methode .attachEvent() aktiviert wurde
 Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)

.fireEvent() ein Event auslösen

.focus() Focus setzen und Focus-Event auslösen
 nur nach dem kompletten Laden des Dokumentes
 vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

.getAdjacentText() Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
 Text kann HTML-Tags enthalten, muss aber nicht



.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementsById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.insertRow()	Zeile in die Tabelle einfügen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.moveRow()	2 Zeilen in der Tabelle austauschen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot

Beispiel:

```

<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
  <TR>
    <TD>Body Zeile 1 Zelle 1</TD>
    <TD>Body Zeile 1 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 2 Zelle 1</TD>
    <TD>Body Zeile 2 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 3 Zelle 1</TD>
    <TD>Body Zeile 3 Zelle 2</TD>
  </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.moveRow(0,1);">verschieben Zeile 0 auf Zeile 1</BUTTON>

```

.normalize()
Normalisierung des DOM zur Erreichung einer konsistenten Struktur



.releaseCapture()	Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5
.setExpression()	Hinweis: ausschalten per Methode .releaseCapture() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.14.1. table.tBody.rows Collection des Internet Explorer

referenziert alle Zeilen der Tabelle im TBODY
laut Reihenfolge der Zeilen im TBODY

ist eine interne Collection, die nur über die Eigenschaft .sectionRowIndex des Objektes table.tr durch Lesen ansprechbar ist (siehe dort):

Diese Eigenschaft liefert den Index der Zeile in dieser Collection.

wird nur erzeugt, wenn TBODY mit Zeilen erzeugt wurde (z.B. per TBODY-Tag)

siehe auch Collection table.rows (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

4.3.2.2.4.3.42.14.2. table.tBody.rows.cells Collection des Internet Explorer

referenziert alle Zellen genau einer Zeile aus TBODY
laut Reihenfolge der Zellen in der Zeile



ist eine interne Collection, die nur über die Collection `table.tBody.rows` ansprechbar ist, welche die Zeile im TBODY indirekt referenziert (siehe dort)

wird nur erzeugt, wenn TBODY mit Zeilen und Zellen erzeugt wurde (z.B. per TBODY-Tag)

Jede Zeile ist eine Element in der Collection `table.tBody.rows`
hat eine eigene Collection `table.tBody.rows.cells`.

siehe auch Collection `table.rows` (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

siehe auch Collection `table.rows.cells` (Indexe aller Zellen einer Zeile der Tabelle, egal wo die Zeile liegt)

4.3.2.2.4.3.42.15. *table.tBodies Collection des Internet Explorer*

referenziert alle TBODY der Tabelle (also nicht THEAD und TFOOT) in der Reihenfolge der TBODY-Elemente in der Tabelle.
Die Erzeugung von TBODY ist im TOM **nicht implementiert**. Dafür ist DOM zu verwenden.

Syntax:

```
[ var ZeigerAufFeld = ] zeiger_auf_tabelle.tBodies
[ var ZeigerAufFeldElement = ] zeiger_auf_tabelle.tBodies[Index]
```

Index Integer, ab 0
muss in [] kodiert werden

zeiger_auf_tabelle laut ID-Attribut

Eigenschaften:

`.length` Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

`.item()` Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit `<INPUT TYPE=image ...>`
da dafür die `children`-Collection verwendet werden muss !

`.namedItem()` Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

`.tags()` Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe `tags` Collection des DOM

`.urns()` Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.42.16. *table.tFoot Objekt des Internet Explorer*

Tabellenfuss TFOOT

es kann maximal nur 1 TFOOT existieren

kann folgende innere Elemente als Kinder haben: TD, TH, TR

Beispiel:

```
<TABLE>
<TBODY>
  <TR>
    <TD>Koerper</TD>
  </TR>
</TBODY>
<TFOOT>
  <TR>
    <TD>Fuss</TD>
  </TR>
</TFOOT>
</TABLE>
```

Eigenschaften:

`.accessKey` Tastaturzugriff auf ein Objekt per Alt + Taste

`.align` Ausrichtung

`ATOMICSELECTION` Selektierbarkeit des Objektes einstellen

`.begin` Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft `.timeAction`
siehe Objekt `currTimeState` und `Behavior` `.style.time2`
deprecated und durch `STYLE`-Attribut zu ersetzen

`.bgColor` prüfen ob Kind möglich ist, also ob Objekt Parent sein kann

`.canHaveChildren` prüfen ob Objekt HTML-Tags enthalten darf

`.canHaveHTML` Klassenreferenz, Klassenname

`.className` Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt

`.clientHeight` Abstand in Pixel zum linken Rand des Fensters

`.clientLeft` Abstand in Pixel zum oberen Rand des Fensters

`.clientTop` Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt

`.clientWidth` Umflussrichtung

`.dir` Objektaktivitäten laut Eigenschaft `.timeAction` beenden

`.end` Zeiger auf das ERSTE Kind laut `childNodes`-Collection eines Objektes

`.firstChild` Objekt ist HTML-Media-Objekt

`.hasMedia` Focussierbarkeit

`.hideFocus` Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)

`.id` ID-Attribut in HTML: Wert ist String alphanumerisch
muss mit Buchstaben beginnen



	Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
	Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
	Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhalte
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie



.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
	"uninitialized" Objekt ist nicht initialisiert
	"loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung
	"loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert
	"interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten
	"complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes
	"middle" zentriert, Standard
	"baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes
	"bottom" am Fuss
	"top" am Kopf

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernen DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus



.cloneNode()	manipuliert nicht den Focus Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.deleteRow()	Zeile löschen aus Tabelle siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName() DOM nicht geändert
.getExpression()	Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.insertRow()	Zeile in die Tabelle einfügen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead



.mergeAttributes()	siehe Objekt table.tFoot alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.moveRow()	2 Zeilen in der Tabelle austauschen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
Beispiel:	<pre> <TABLE ID="ID_Tabelle" BORDER CELSPACING=10> <TR> <TD>Body Zeile 1 Zelle 1</TD> <TD>Body Zeile 1 Zelle 2</TD> </TR> <TR> <TD>Body Zeile 2 Zelle 1</TD> <TD>Body Zeile 2 Zelle 2</TD> </TR> <TR> <TD>Body Zeile 3 Zelle 1</TD> <TD>Body Zeile 3 Zelle 2</TD> </TR> </TABLE> <BUTTON onclick="ID_Tabelle.moveRow(0,1);">verschieben Zeile 0 auf Zeile 1</BUTTON> </pre>
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern



DOM wird geändert
 Maus-Überwachung einschalten für ein Objekt
 Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
 ab IE 5.5
 Hinweis: ausschalten per Methode .releaseCapture()
 Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.4.3.42.16.1. *table.tFoot.rows* Collection des Internet Explorer

referenziert alle Zeilen der Tabelle im TFOOT
 laut Reihenfolge der Zeilen im TFOOT

ist eine interne Collection, die nur über die Eigenschaft .sectionRowIndex des Objektes table.tr durch Lesen ansprechbar ist (siehe dort):

Diese Eigenschaft liefert den Index der Zeile in dieser Collection.

wird nur erzeugt, wenn TFOOT mit Zeilen erzeugt wurde (z.B. per TFOOT-Tag)

siehe auch Collection table.rows (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

4.3.2.2.4.3.42.16.2. *table.tFoot.rows.cells* Collection des Internet Explorer

referenziert alle Zellen genau einer Zeile aus TFOOT
 laut Reihenfolge der Zellen in der Zeile

ist eine interne Collection, die nur über die Collection table.tFoot.rows ansprechbar ist, welche die Zeile im TFOOT indirekt referenziert (siehe dort)

wird nur erzeugt, wenn TFOOT mit Zeilen und Zellen erzeugt wurde (z.B. per TFOOT-Tag)

Jede Zeile ist eine Element in der Collection table.tFoot.rows
 hat eine eigene Collection table.tFoot.rows.cells.

siehe auch Collection table.rows (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

siehe auch Collection table.rows .cells (Indexe aller Zellen einer Zeile der Tabelle, egal wo die Zeile liegt)

4.3.2.2.4.3.42.17. *table.tHead* Objekt des Internet Explorer

Tabellenkopf THEAD

es kann maximal nur 1 THEAD existieren

kann folgende innere Elemente als Kinder haben: TD, TH, TR

Beispiel:

```
<TABLE>
<THEAD>
  <TR>
    <TD>Kopf</TD>
  </TR>
</THEAD>
<TBODY>
  <TR>
    <TD>Body</TD>
  </TR>
</TBODY>
</TABLE>
```

Eigenschaften:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste
 .align Ausrichtung
 ATOMICSELECTION Selektierbarkeit des Objektes einstellen
 .begin Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
 siehe Objekt currTimeState und Behavior .style.time2
 .bgColor deprecated und durch STYLE-Attribut zu ersetzen
 .canHaveChildren prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
 .canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf
 .className Klassenreferenz, Klassenname
 .clientHeight Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
 .clientLeft Abstand in Pixel zum linken Rand des Fensters
 .clientTop Abstand in Pixel zum oberen Rand des Fensters
 .clientWidth Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
 .dir Umflussrichtung



.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id); Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isContentEditable	Interaktionsfähigkeit
.isDisabled	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt



	nur nach kompletten einlesen des Dokumentes nutzbar
	nur Plain-Text
	schreiben: immer komplett überschreiben
	nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
	"uninitialized" Objekt ist nicht initialisiert
	"loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung
	"loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert
	"interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten
	"complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes
	immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes
	nur nutzbar nach dem kompletten Laden des Dokumentes
	immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style)
	Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes
	siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes
	Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde
	kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes
	"middle" zentriert, Standard
	"baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes
	"bottom" am Fuss
	"top" am Kopf
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen
	Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
	ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
	DOM wird geändert
	Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
	Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
	DOM wird geändert
	Element kann selbst Kinder haben
	Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
	Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler
	Hinweis: Abschalten mit Methode .detachEvent()
	Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen
	Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !



	vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.deleteRow()	Zeile löschen aus Tabelle siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein



	Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.insertRow()	Zeile in die Tabelle einfügen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu genießen !! DOM wird geändert
.moveRow()	2 Zeilen in der Tabelle austauschen siehe Objekt table siehe Objekt table.tBody siehe Objekt table.tHead siehe Objekt table.tFoot
Beispiel:	<pre> <TABLE ID="ID_Tabelle" BORDER CELSPACING=10> <TR> <TD>Body Zeile 1 Zelle 1</TD> <TD>Body Zeile 1 Zelle 2</TD> </TR> <TR> <TD>Body Zeile 2 Zelle 1</TD> <TD>Body Zeile 2 Zelle 2</TD> </TR> <TR> <TD>Body Zeile 3 Zelle 1</TD> <TD>Body Zeile 3 Zelle 2</TD> </TR> </TABLE> <BUTTON onclick="ID_Tabelle.moveRow(0,1);">verschieben Zeile 0 auf Zeile 1</BUTTON> </pre>
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein



<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode <code>.releaseCapture()</code>
<code>.setExpression()</code>	Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft.</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.17.1. *table.tHead.rows* Collection des Internet Explorer

referenziert alle Zeilen der Tabelle im THEAD
laut Reihenfolge der Zeilen im THEAD

ist eine interne Collection, die nur über die Eigenschaft `.sectionRowIndex` des Objektes `table.tr` durch Lesen ansprechbar ist (siehe dort):

Diese Eigenschaft liefert den Index der Zeile in dieser Collection.

wird nur erzeugt, wenn THEAD mit Zeilen erzeugt wurde (z.B. per THEAD-Tag)

siehe auch Collection `table.rows` (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

4.3.2.2.4.3.42.17.2. *table.tHead.rows.cells* Collection des Internet Explorer

referenziert alle Zellen genau **einer** Zeile aus THEAD
laut Reihenfolge der Zellen in der Zeile

ist eine interne Collection, die nur über die Collection `table.tHead.rows` ansprechbar ist, welche die Zeile im THEAD indirekt referenziert (siehe dort)

wird nur erzeugt, wenn THEAD mit Zeilen und Zellen erzeugt wurde (z.B. per THEAD-Tag)

Jede Zeile ist eine Element in der Collection `table.tHead.rows`
hat eine eigene Collection `table.tHead.rows.cells`.

siehe auch Collection `table.rows` (Indexe aller Zeilen der Tabelle, egal wo sie liegen)

siehe auch Collection `table.rows.cells` (Indexe aller Zellen einer Zeile der Tabelle, egal wo die Zeile liegt)

4.3.2.2.4.3.42.18. *table.tr* Objekt des Internet Explorer

Tabellenzeile TR

enthält alle Zellen der Zeile

Die Zeilen innerhalb der Tabelle werden in der Collection `table.rows` referenziert, wobei der Index der Collection die Zeilennummer ab 0 darstellt.

pro Zeile eine eigene Collection `table.rows.cells`

Beispiel 1:

```
<TABLE>
  <TR>
    <TD>Zelle1</TD>
  </TR>
  <TR>
    <TD>Zelle2</TD>
  </TR>
</TABLE>
```

Beispiel 2:

```
<SCRIPT>
function ZeilenErzeugen()
{
    // +++++ Collection aller Tabellenzeilen adressieren
    var TabellenZeilenCollection = ID_Tabelle.rows;

    // +++++ Zeile 1 erzeugen durch anhängen
    // ----- aktuelle Anzahl der Tabellenzeilen ermitteln
```




```

var AnzahlTabellenZeilen = TabellenZeilenCollection.length;
// ----- Zeile 1 in Tabelle einbinden sowie der Collection anhängen
var TabellenZeile1 = ID_Tabelle.insertRow(AnzahlTabellenZeilen);

// ++++++ Zeile 2 erzeugen durch anhängen
// ----- aktuelle Anzahl der Tabellenzeilen ermitteln
AnzahlTabellenZeilen = TabellenZeilenCollection.length;
// ----- Zeile 2 in Tabelle einbinden sowie der Collection anhängen
var TabellenZeile2= ID_Tabelle.insertRow(AnzahlTabellenZeilen);

// ++++++ Zeile 1 mit 2 Zellen versorgen
// ----- Collection der Zellen adressieren
var TabellenZeile_ZellenCollection = TabellenZeile1.cells;

// ----- Zeile 1 in die Tabellenzeile 1 einfügen und der Collection anhängen
var AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
var AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
var AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile1_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ----- Zeile 2 in die Tabellenzeile 1 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile1_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ++++++ Zeile 2 mit 2 Zellen versorgen
// ----- Collection der Zellen adressieren
var TabellenZeile_ZellenCollection = TabellenZeile2.cells;

// ----- Zeile 1 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile2_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ----- Zeile 2 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile2_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// alle Zellen mit Daten versorgen, die sofort angezeigt werden (kein Tabellen-Refresh nötig)
TabelleZeile1_Zelle1.innerHTML="<B>Zeile1 Zeile1</B>";
TabelleZeile1_Zelle2.innerHTML="<B>Zeile1 Zeile2</B>";
TabelleZeile2_Zelle1.innerHTML="<B>Zeile2 Zeile1</B>";
TabelleZeile2_Zelle2.innerHTML="<B>Zeile2 Zeile2</B>";
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Zeilen erzeugen" onclick="ZeilenErzeugen();">
<TABLE ID="ID_Tabelle" BORDER=1>
</TABLE>

```

Eigenschaften:

Folgende Eigenschaften der Zeile liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index des Zeile im DOM
.rowIndex	Index der Zeile bezüglich Tabelle
.sectionRowIndex	Index der Zeile bezüglich Tabellenelement wie THEAD, TBODY bzw. TFOOT nur für TD, nicht für TH

Folgende Eigenschaften der Zelle liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index der Zelle im DOM
.cellIndex	Index der Zelle innerhalb der Zeile
.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.borderColor	Borderfarbe (Rahmenfarbe) wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no" Eigenschaft .border mit Wert 0
#rrggbb	vordefinierter Farbname (browserspezifisch)



.borderColorDark	deprecated und durch Eigenschaft .borderColor zu ersetzen dunkle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.borderColorLight	deprecated und durch Eigenschaft .borderColor zu ersetzen helle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich true Element nicht interaktionsfähig false Default, Element ist interaktionfähig
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id); Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE , TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)



.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.rowIndex	Index der Tabellenzeile in der Collection table.rows
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sectionRowIndex	Index der Tabellenzeile in der Collection table.tBody.rows bzw. table.tFoot.rows bzw. table.tHead.rows Zeile muss im existierenden Tabellenteil TOBODY bzw. TFOOT bzw. THEAD liegen
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf



.width	Breite des Objektes in Pixel Integer in Pixels für absolute Breite, >=0 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel String Ziffernfolge eines Integer mit nachfolgendem % für Breite als Anteil der Breite des Elternobjektes z.B. 10%
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.deleteCell()	Zelle in die Zeile einer Tabelle löschen
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster



	Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByTagName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.insertCell()	Zelle TD in die Zeile einer Tabelle einfügen Zelle TH nicht direkt erzeugbar: Es muss .innerHTML mit -Tag gefüllt werden
Beispiel :	
	var Zeile = zeiger_auf_tabelle.insertRow(); var Zelle = Zeile.insertCell(); Zelle.innerHTML = "<I>Test </I>";
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein



	Sichtbarkeit erst wenn Ende-Tag geparkt wurde
	DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endtags
	DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
<code>.setExpression()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient
	Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endtag geparkt DOM wird geändert

4.3.2.2.4.3.42.18.1. *table.tr.td* Objekt des Internet Explorer

Zelle TD einer Tabellenzeile

enthält die Daten

Beispiel 1:

```
<TABLE>
  <TR>
    <TD>Zelle1</TD>
  </TR>
  <TR>
    <TD>Zelle2</TD>
  </TR>
</TABLE>
```

Beispiel 2:

```
<SCRIPT>
function ZeilenErzeugen()
{
    //+++++ Collection aller Tabellenzeilen adressieren
    var TabellenZeilenCollection = ID_Tabelle.rows;

    //+++++ Zeile 1 erzeugen durch anhängen
    //----- aktuelle Anzahl der Tabellenzeilen ermitteln
    var AnzahlTabelleZeilen = TabellenZeilenCollection.length;
    //----- Zeile 1 in Tabelle einbinden sowie der Collection anhängen
    var TabellenZeile1 = ID_Tabelle.insertRow(AnzahlTabelleZeilen);

    //+++++ Zeile 2 erzeugen durch anhängen
    //----- aktuelle Anzahl der Tabellenzeilen ermitteln
    AnzahlTabelleZeilen = TabellenZeilenCollection.length;
    //----- Zeile 2 in Tabelle einbinden sowie der Collection anhängen
    var TabellenZeile2 = ID_Tabelle.insertRow(AnzahlTabelleZeilen);

    //+++++ Zeile 1 mit 2 Zellen versorgen
    //----- Collection der Zellen adressieren
    var TabellenZeile_ZellenCollection = TabellenZeile1.cells;

    //----- Zeile 1 in die Tabellenzeile 1 einfügen und der Collection anhängen
    var AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
    var AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
    var AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
    var TabellenZeile1_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

    //----- Zeile 2 in die Tabellenzeile 1 einfügen und der Collection anhängen
    AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
    AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
```




```

AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile1_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// +++++ Zeile 2 mit 2 Zellen versorgen
// ----- Collection der Zellen adressieren
var TabellenZeile_ZellenCollection = TabellenZeile2.cells;

// ----- Zeile 1 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile2_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ----- Zeile 2 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile_ZellenCollection.length;
var TabellenZeile2_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// alle Zellen mit Daten versorgen, die sofort angezeigt werden (kein Tabellen-Refresh nötig)
Tabelle1_Zelle1.innerHTML="Zeile1 Zeile1";
Tabelle1_Zelle2.innerHTML="Zeile1 Zeile2";
Tabelle2_Zelle1.innerHTML="Zeile2 Zeile1";
Tabelle2_Zelle2.innerHTML="Zeile2 Zeile2";
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Zeilen erzeugen" onclick="ZeilenErzeugen();">
<TABLE ID="ID_Tabelle" BORDER=1>
</TABLE>

```

Eigenschaften:

Eigenschaften .innerText und .innerHTML sind les- und schreibbar
Eigenschaft .style verfügbar

Folgende Eigenschaften der Zelle liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index der Zelle im DOM
.cellIndex	Index der Zelle innerhalb der Zeile

Folgende Eigenschaften der Zeile liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index der Zeile im DOM
.rowIndex	Index der Zeile bezüglich Tabelle
.sectionRowIndex	Index der Zeile bezüglich Tabellenelement wie THEAD, TBODY bzw. TFOOT nur für TD, nicht für TH

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.background	Hintergrundbild eines Objektes
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.borderColor	Borderfarbe (Rahmenfarbe) wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no" Eigenschaft .border mit Wert 0
	#rrggbb vordefinierter Farbname (browserspezifisch)
.borderColorDark	deprecated und durch Eigenschaft .borderColor zu ersetzen dunkle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.borderColorLight	deprecated und durch Eigenschaft .borderColor zu ersetzen helle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.cellIndex	Index der Zelle in der Collection table.rows.cells siehe Objekt table.tr.td siehe Objekt table.tr.th
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.colSpan	Anzahl der Spalten einer Zelle in einer Tabelle siehe Objekt table.tr.td siehe Objekt table.tr.th



.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich true Element nicht interaktionsfähig false Default, Element ist interaktionsfähig
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id); Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag nur Plain-Text also keine HTML-Elemente und kein Script schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.noWrap	Wortumbruch einstellen false Default. Browser bricht den Text automatisch um true Browser bricht den Text nicht um
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x



.outerHTML	Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.rowSpan	Anzahl der Zeilen einer Zelle in einer Tabelle siehe Objekt table.tr.td siehe Objekt table.tr.th
.scope	Gruppierung, in der eine Zelle liegt siehe Objekt table.tr.td siehe Objekt table.tr.th "row" Zelle enthält Header-Informationen zur Zeile "col" Zelle enthält Header-Informationen zur Spalte "rowgroup" wie "row" aber zur Gruppe aus Zeilen "colgroup" wie "col" aber zur Gruppe aus Spalten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sectionRowIndex	Index der Tabellenzeile in der Collection table.tBody.rows bzw. table.tFoot.rows bzw. table.tHead.rows
.sourceIndex	Zeile muss im existierenden Tabellenteil TOBODY bzw. TFOOT bzw. THEAD liegen
STYLE	Index des Objektes in der Collection document.all direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes "off" Default. selektierbar "on" nicht selektierbar
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des



benachbarten Objektes

"bottom" am Fuss
 "top" am Kopf
 .width Breite des Objektes in Pixel
 Integer in Pixels für absolute Breite, >=0
 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel
 String Ziffernfolge eines Integer mit nachfolgendem % für Breite als
 Anteil der Breite des Elternobjektes
 z.B. 10%

Methoden:

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen
 Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
 ab IE 5.x bis unter IE 5.5

.appendChild() Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
 DOM wird geändert
 Zeiger wird zugleich immer am Ende der Collection childNodes angehängen
 Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde

.applyElement() Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
 DOM wird geändert
 Element kann selbst Kinder haben
 Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde
 Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in **Zufallsfolge**, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)

.blur() Element den Focus wegnehmen und Event onblur auslösen
 Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
 vor IE 5.0 TABINDEX-Attribut muss kodiert sein
 ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein

.clearAttributes() alle HTML-Attribute eines Objektes entfernen
 außer ID, STYLE und per Script definierte Attribute
 Script-erzeugte Attribute nicht entfernbar
 DOM wird geändert

.click() simuliert einen Klick auf das Element und löst onclick-Event aus
 manipuliert nicht den Focus

.cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
 DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)

.componentFromPoint() Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
 auch für CSS-Layout
 onmouseover-Event hat **nicht die Pixelgenauigkeit** wie die Angaben der Methode .componentFromPoint()
 also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelposition erreicht haben
 Overbereich der Maus ist mehr als 1 Pixel gross
 beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.

.contains() prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
 DOM nicht geändert

.detachEvent() Abschalten des Registrieren eines Events durch Eventhandler
 wobei Registrierung mit Methode .attachEvent() aktiviert wurde
 Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)

.dragDrop() prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element

.fireEvent() ein Event auslösen

.focus() Focus setzen und Focus-Event auslösen
 nur nach dem kompletten Laden des Dokumentes
 vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

.getAdjacentText() Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
 Text kann HTML-Tags enthalten, muss aber nicht
 DOM nicht geändert

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
 DOM nicht geändert

.getAttributeNode() Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft.
 Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht
 Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt
 Wert des Attributes wird somit über die Referenz laut DOM erreichbar
 DOM nicht geändert



<code>.getBoundingClientRect()</code>	Referenz auf TextRectangle-Objekt im Element holen
<code>.getClientRects()</code>	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
<code>.getElementsByTagName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.getExpression()</code>	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden <code>expression()</code> oder <code>setExpression()</code> zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
<code>.hasChildNodes()</code>	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . Hinweis: einschalten per Methode <code>.setCapture()</code>
<code>.removeAttribute()</code>	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde



<code>.replaceAdjacentText()</code>	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
<code>.replaceChild()</code>	DOM wird nicht geändert Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde
<code>.replaceNode()</code>	DOM wird geändert Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags
<code>.scrollIntoView()</code>	DOM wird geändert Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
<code>.setExpression()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> . dient Ausdruck nur als Script kodierbar
<code>.swapNode()</code>	DOM wird nicht geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.42.18.2. *table.tr.th* Objekt des Internet Explorer

Zelle TH einer Tabellenzeile

enthält keine Daten

dient nur der Spaltenüberschrift (Inhalt der Zelle) in automatischer Fettdarstellung

als Objekt nicht per Script erzeugbar:

Erzeugung nur als TD-Zelle, wobei `.innerHTML` mit ``-Tag gefüllt werden muss

Beispiel 1:

```
<TABLE>
  <TR>
    <TH>Zelle1 fett</TH>
  </TR>
  <TR>
    <TH>Zelle2 fett</TH>
  </TR>
</TABLE>
```

Beispiel 2:

```
<SCRIPT>
function ZeilenErzeugen()
{
    // +++++ Collection aller Tabellenzeilen adressieren
    var TabellenZeilenCollection = ID_Tabelle.rows;

    // +++++ Zeile 1 erzeugen durch anhängen
    // ----- aktuelle Anzahl der Tabellenzeilen ermitteln
    var AnzahlTabelleZeilen = TabellenZeilenCollection.length;
    // ----- Zeile 1 in Tabelle einbinden sowie der Collection anhängen
    var TabellenZeile1 = ID_Tabelle.insertRow(AnzahlTabelleZeilen);

    // +++++ Zeile 2 erzeugen durch anhängen
    // ----- aktuelle Anzahl der Tabellenzeilen ermitteln
    AnzahlTabelleZeilen = TabellenZeilenCollection.length;
    // ----- Zeile 2 in Tabelle einbinden sowie der Collection anhängen
    var TabellenZeile2 = ID_Tabelle.insertRow(AnzahlTabelleZeilen);

    // +++++ Zeile 1 mit 2 Zellen versorgen
    // ----- Collection der Zellen adressieren
    var TabellenZeile_ZellenCollection = TabellenZeile1.cells;
```



```

// ----- Zelle 1 in die Tabellenzeile 1 einfügen und der Collection anhängen
var AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
var AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
var AktuelleAnzahlZellen = TabellenZeile1.ZellenCollection.length;
var TabellenZeile1_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ----- Zelle 2 in die Tabellenzeile 1 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile1.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile1.ZellenCollection.length;
var TabellenZeile1_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// +++++ Zeile 2 mit 2 Zellen versorgen
// ----- Collection der Zellen adressieren
var TabellenZeile_ZellenCollection = TabellenZeile2.cells;

// ----- Zelle 1 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile2.ZellenCollection.length;
var TabellenZeile2_Zelle1 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// ----- Zelle 2 in die Tabellenzeile 2 einfügen und der Collection anhängen
AktuellerIndexDerZelle = TabellenZeile2.rowIndex;
AktuelleZelle = TabellenZeilenCollection(AktuellerIndexDerZelle);
AktuelleAnzahlZellen = TabellenZeile2.ZellenCollection.length;
var TabellenZeile2_Zelle2 = AktuelleZelle.insertCell(AktuelleAnzahlZellen);

// alle Zellen mit Daten versorgen, die sofort angezeigt werden (kein Tabellen-Refresh nötig)
Tabelle1_Zelle1.innerHTML="<B>Zeile1 Zelle1</B>";
Tabelle1_Zelle2.innerHTML="<B>Zeile1 Zelle2</B>";
Tabelle2_Zelle1.innerHTML="<B>Zeile2 Zelle1</B>";
Tabelle2_Zelle2.innerHTML="<B>Zeile2 Zelle2</B>";
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Zeilen erzeugen" onclick="ZeilenErzeugen();">
<TABLE ID="ID_Tabelle" BORDER=1>
</TABLE>

```

Eigenschaften:

Eigenschaften .innerText und .innerHTML sind les- und schreibbar
Eigenschaft .style verfügbar

Folgende Eigenschaften der Zelle liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index der Zelle im DOM
.cellIndex	Index der Zelle innerhalb der Zeile

Folgende Eigenschaften der Zeile liefern Indexe im DOM bzw. TOM:

.sourceIndex	Index der Zeile im DOM
.rowIndex	Index der Zeile bezüglich Tabelle
.sectionRowIndex	Index der Zeile bezüglich Tabellenelement wie THEAD, TBODY bzw. TFOOT nur für TD, nicht für TH

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.background	Hintergrundbild eines Objektes
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.borderColor	Borderfarbe (Rahmenfarbe) wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no" Eigenschaft .border mit Wert 0
	#rrggbb vordefinierter Farbname (browserspezifisch)
.borderColorDark	deprecated und durch Eigenschaft .borderColor zu ersetzen dunkle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.borderColorLight	deprecated und durch Eigenschaft .borderColor zu ersetzen helle Farbe des 3D-Rahmens eines Objektes zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.cellIndex	Index der Zelle in der Collection table.rows.cells siehe Objekt table.tr.td



	siehe Objekt table.tr.th
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
.colSpan	Anzahl der Spalten einer Zelle in einer Tabelle
	siehe Objekt table.tr.th
.dir	Umflussrichtung
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
	ID-Attribut in HTML: Wert ist String
	alphanumerisch
	muss mit Buchstaben beginnen
	Unterstrich _ verwendbar
	kann in " " bzw. ' ' kodiert werden, muss aber nicht
	Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
	Zeiger aus ID bilden var Zeiger = eval(object.id);
	Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
	Plain-Text
	HTML-Elemente je nach Objekt möglich
	Script: muss mit DEFER-Attribut kodiert sein
	schreiben: wenn Bereich nicht leer, so komplett überschreiben
	wenn Bereich leer so einfügen
	nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
	ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
	dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
	also nach dem Laden des Objektes
	aber wirksam erst mit parsen des HTML-Endetags
	nur Plain-Text also keine HTML-Elemente und kein Script
	schreiben: wenn Bereich nicht leer, so komplett überschreiben
	wenn Bereich leer so einfügen
	nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit
	nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element)
	also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
	1 für Element-Knoten.
	3 für Textknoten.
	oder null (nicht 0 !!) wenn Knoten nicht vorhanden
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes)
	nur für Text- und Attribut-Elemente
	nicht für Element-Knoten (Knotentyp 1)
.noWrap	Wortumbruch einstellen
	false Default.
	Browser bricht den Text automatisch um
	true Browser bricht den Text nicht um
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern
	für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth



.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar nur Plain-Text schreiben: immer komplett überschreiben nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.rowSpan	Anzahl der Zeilen einer Zelle in einer Tabelle siehe Objekt table.tr.td siehe Objekt table.tr.th
.scope	Gruppierung, in der eine Zelle liegt siehe Objekt table.tr.td siehe Objekt table.tr.th "row" Zelle enthält Header-Informationen zur Zeile "col" Zelle enthält Header-Informationen zur Spalte "rowgroup" wie "row" aber zur Gruppe aus Zeilen "colgroup" wie "col" aber zur Gruppe aus Spalten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf



.width	<p>Breite des Objektes in Pixel</p> <p>Integer in Pixels für absolute Breite, >=0</p> <p>bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel</p> <p>String Ziffernfolge eines Integer mit nachfolgendem % für Breite als Anteil der Breite des Elternobjektes</p> <p>z.B. 10%</p>
Methoden:	
.addBehavior()	<p>DHTML-Verhaltenseigenschaft einem Element hinzufügen</p> <p>Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).</p> <p>ab IE 5.x bis unter IE 5.5</p>
.appendChild()	<p>Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern</p> <p>DOM wird geändert</p> <p>Zeiger wird zugleich immer am Ende der Collection childNodes angehängen</p> <p>Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde</p>
.applyElement()	<p>Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern</p> <p>DOM wird geändert</p> <p>Element kann selbst Kinder haben</p> <p>Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde</p> <p>Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !</p>
.attachEvent()	<p>Einschalten des Registrieren eines Events durch Eventhandler</p> <p>Hinweis: Abschalten mit Methode .detachEvent()</p> <p>Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)</p>
.blur()	<p>Element den Focus wegnehmen und Event onblur auslösen</p> <p>Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !</p> <p>vor IE 5.0 TABINDEX-Attribut muss kodiert sein</p> <p>ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein</p>
.clearAttributes()	<p>alle HTML-Attribute eines Objektes entfernen</p> <p>außer ID, STYLE und per Script definierte Attribute</p> <p>Script-erzeugte Attribute nicht entfernbar</p> <p>DOM wird geändert</p>
.click()	<p>simuliert einen Klick auf das Element und löst onclick-Event aus</p> <p>manipuliert nicht den Focus</p>
.cloneNode()	<p>Objekt klonen und Referenz des erzeugten Klone liefern</p> <p>DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)</p>
.componentFromPoint()	<p>Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt</p> <p>auch für CSS-Layout</p> <p>onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben</p> <p>Overbereich der Maus ist mehr als 1 Pixel gross</p> <p>beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.</p>
.contains()	<p>prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist</p> <p>DOM nicht geändert</p>
.detachEvent()	<p>Abschalten des Registrieren eines Events durch Eventhandler</p> <p>wobei Registrierung mit Methode .attachEvent() aktiviert wurde</p> <p>Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)</p>
.fireEvent()	<p>ein Event auslösen</p>
.focus()	<p>Focus setzen und Focus-Event auslösen</p> <p>nur nach dem kompletten Laden des Dokumentes</p> <p>vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen</p>
.getAdjacentText()	<p>Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann</p> <p>Text kann HTML-Tags enthalten, muss aber nicht</p> <p>DOM nicht geändert</p>
.getAttribute()	<p>Wert eines per HTML erzeugten Attributes liefern</p> <p>DOM nicht geändert</p>
.getAttributeNode()	<p>Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft.</p> <p>Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht</p> <p>Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt</p> <p>Wert des Attributes wird somit über die Referenz laut DOM erreichbar</p> <p>DOM nicht geändert</p>
.getBoundingClientRect()	<p>Referenz auf TextRectangle-Objekt im Element holen</p>
.getClientRects()	<p>Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster</p> <p>Feld mit Index als Integer ab 0</p> <p>pro Eintrag ein Rectangle</p>



<code>.getElementsByTagName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.getExpression()</code>	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
<code>.hasChildNodes()</code>	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.insertAdjacentHTML()</code>	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <code><SCRIPT DEFER></code> muss kodiert werden DOM wird geändert
<code>.insertAdjacentText()</code>	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
<code>.insertBefore()</code>	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode <code>createElement()</code> erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.releaseCapture()</code>	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
<code>.removeAttribute()</code>	Hinweis: einschalten per Methode <code>.setCapture()</code> entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode <code>.createAttribute()</code> erzeugte Attribute werden nicht erfasst DOM wird geändert
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.removeChild()</code>	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
<code>.removeExpression()</code>	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> dient. Ausdruck muss mit der Methode <code>.setExpression()</code> gesetzt worden sein DOM wird nicht geändert
<code>.removeNode()</code>	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceAdjacentText()</code>	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert



<code>.replaceChild()</code>	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode <code>.createElement()</code> erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
<code>.replaceNode()</code>	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
<code>.scrollIntoView()</code>	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
<code>.setActive()</code>	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
<code>.setAttribute()</code>	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
<code>.setCapture()</code>	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : <code>onmousedown</code> , <code>onmouseup</code> , <code>onmousemove</code> , <code>onclick</code> , <code>ondblclick</code> , <code>onmouseover</code> und <code>onmouseout</code> . ab IE 5.5
<code>.setExpression()</code>	Hinweis: ausschalten per Methode <code>.releaseCapture()</code> Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft</code> . dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
<code>.swapNode()</code>	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.43. textarea Objekt des Internet Explorer

Objekt des mehrzeiligen Text-Eingabe-Controls (in HTML der Tag TEXTAREA)

Beispiel 1:

```
<FORM ACTION="mailto:test@test.de" METHOD=GET>
  <INPUT NAME=subject TYPE=hidden
    VALUE="Testt%20Product%20Information%20Anforderung"
  >
  volle Mailadresse eingeben
  <BR>
  <TEXTAREA NAME=body COLS=40></TEXTAREA>
  <INPUT TYPE=submit VALUE="absenden "
</FORM>

<SCRIPT>
function Anzeige(ZeigerAufTextarea)
{
    var FeldAllterTextArea = document.all.tags("TEXTAREA");

    // prüfen ob mindestens 1 TEXTAREA-Element im Body vorliegt
    if (FeldAllterTextArea != null)
    {
        var TextBereich = ZeigerAufTextarea.createTextRange();

        // prüfen ob Textbereich erzeugt wurde
        if (TextBereich != null)
        { alert("BoundingHeight = " + TextBereich.boundingHeight); }
    }
}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS=100 ROWS=2 onclick="Anzeige(this)">
.....
</TEXTAREA>
```

Beispiel 2:

```
<SCRIPT>
function HoleHTML()
{ ID_Textarea.value= document.documentElement.innerHTML; }
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS = 50 ROWS = 10>
</TEXTAREA>
```

Beispiel 3:



```

<HTML>
<HEAD>
<SCRIPT>
    function Antworten(Wert)
    {Wert ? alert("Ja") : alert("Nein");}
</SCRIPT>
</HEAD>
<BODY>
    <P>
        <INPUT type="text" ID="ID_Input" VALUE="Test">
        <BUTTON onclick="Antworten(ID_Input.isMultiLine);">
            Mehrzeiligkeit eines INPUT TYPE=text
        </BUTTON>
    </P>
    <P>
        TEXTAREA:
        <TEXTAREA ID="ID_Textarea">
            Test
        </TEXTAREA>
        <BUTTON onclick="Antworten(ID_Textarea.isMultiLine);">
            Mehrzeiligkeit eines Textbereiches
        </BUTTON>
    </P>
</BODY>
</HTML>

```

Beispiel 4:

```

<SPAN UNSELECTABLE="on" >
    Dieser Text kann nicht selektiert werden
    <TEXTAREA WRAP="PHYSICAL" ROWS="5"
        STYLE="font-weight: bold;"
    >
        Dieser Text kann selektiert werden
    </TEXTAREA>
    Dieser Text kann nicht selektiert werden
</SPAN>

```

Beispiel 5:

```

<HEAD>
<SCRIPT>
    function ScrolleSeiteRechts()
    {document.body.doScroll("scrollbarPageRight");}

    function ScrolleSeiteDown()
    {ID_Textaerea.doScroll("scrollbarPageDown");}
</SCRIPT>
</HEAD>
<BODY>
    <BUTTON onclick=" ScrolleSeiteRechts()">
        scrolle Seite rechts
    </BUTTON>
    <BR>
    <BUTTON
        onclick=" ScrolleSeiteDown()"
        ondblclick=" ScrolleSeiteDown()"
    >
        Texarea: Click = scrolle down Doppelklick = scrolle Seite down
    </BUTTON>
    <BR>
    <BR>
    <TEXTAREA ID="ID_Textaerea" >
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
    </TEXTAREA>
</BODY>

```



Beispiel 6:

```

<BODY onload="ID_Div.setCapture();"
onlick="document.releaseCapture();"
>
  <DIV ID="ID_Div"
    onmousemove="ID_Textarea.value = event.clientX + event.clientY;"
    onlosecapture="alert(event.srcElement.id + ' hat keine Mausueberwachung mehr');">
    <TEXTAREA ID="ID_Textarea" COLS=2>Test</TEXTAREA>
  </DIV>
</BODY>

<HTML>
<HEAD>
<SCRIPT>
  function EventHandler_AktionVorReset()
  {
    // ein Hinweis im Textarea anzeigen
    ID_Textarea.value += "Formular zuruecksetzen ????";

    // wirklich rücssetzen ???
    return( confirm("Wirklich rücssetzen ?"));
    // true so Reset durch Browser ausführen lassen
    // false so kein Reset durch Browser
  }
</SCRIPT>
</HEAD>
<BODY>
  <DIV>
    <FORM NAME="Formular" onreset="EventHandler_AktionVorReset();">
      <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
      <BR>
      <BUTTON onclick="form.reset();">OnReset ausloesen</BUTTON>
      <BR><BR>
      <INPUT TYPE="reset" VALUE="oder hiermit zuruecksetzen"
        onclick="form.reset()"
      >
    </FORM>
  </DIV>
  <TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst
ATOMICSELECTION	vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.cols	Anzahl der sichtbaren Zeichen in einer Zeile der TEXTAREA Hinweis: Es können mehr Zeichen in der TEXTAREA enthalten sein als sichtbar siehe .rows für Mehrzeiligkeit .wrap für Wortumbruch
.contentEditable	Scrollleisten sind erzeugbar Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.defaultValue	Vorbelegung der TEXTAREA



	sichtbare Auswirkungen nur bei Instanzierung des Objektes Formular-Reset
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readOnly	Editierbarkeit eines Text-Controls Editierung bedeutet Focuserhalt ! false Default Objekt ist nicht read-only also ist es editierbar kann also Focus erhalten true Objekt ist read-only



	also ist es nicht editierbar kann kein also Focus erhalten
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.rows	Anzahl der sichtbaren Zeilen der TEXTAREA
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.status	Selektionsstatus eines Control-Elementes Control-Element: kann durch User interaktiv verändert werden wird bei Ceckbox-Control auch verändert durch Eigenschaft .indeterminate bei Statusveränderung wird Event onpropertychange erzeugt false Default außer bei textArea Objekt Control ist nicht selektiert true Control ist selektiert null Control ist nicht initialisiert Default bei textArea Objekt
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLETT, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.type	Type des TEXTAREA-Control
.uniqueID	durch den Browser automatisch-geniertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar
.wrap	Wortumbruch der TEXTAREA "soft" Standard Wortumbruch aktiv im Formular wird nicht gesendet: Zeilenumbruch Zeilenvorschub "hard" Wortumbruch aktiv im Formular wird gesendet: Zeilenumbruch Zeilenvorschub "off" Wortumbruch nicht aktiv



Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endtag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
.createTextRange()	DOM nicht geändert Textbereich erzeugen

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
var FeldAllerButtonElemente coll = document.all.tags("BUTTON");

if ( (FeldAllerButtonElemente !=null )
    && (FeldAllerButtonElemente.length>0)
    )
{
    var ZeigerAufRange = FeldAllerButtonElemente[0].createTextRange();
    ZeigerAufRange.text = "Clickted";
}
</SCRIPT>

```

Beispiel 2:

```

function ErzeugeUndSelektiereTextRange()
{
    var TextBereich = document.body.createTextRange();
    TextBereich.findText("Testtext");
    TextBereich.select();
}

```

.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichnung zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Klick auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein !



"scrollbarDown"	oder "down"	Down scroll Pfeil
"scrollbarHThumb"		horizontaler Scrollbalken
"scrollbarLeft"	oder "left"	Left scroll Pfeil
"scrollbarPageDown"	oder "pageDown"	Page-down Scrollbalken
"scrollbarPageLeft"	oder "pageLeft"	Page-left Scrollbalken
"scrollbarPageRight"	oder "pageRight"	Page-right Scrollbalken
"scrollbarPageUp"	oder "pageUp"	Page-up Scrollbalken
"scrollbarRight"	oder "right"	Right scroll Pfeil
"scrollbarUp"	oder "up"	Up scroll Pfeil
"scrollbarVThumb"		vertikaler Scrollbalken

Beispiel:

```
<HEAD>
<SCRIPT>
function ScrolleSeiteRechts()
{ document.body.doScroll("scrollbarPageRight");}

function ScrolleDown()
{ID_Textaerea.doScroll("scrollbarDown");}

function ScrolleSeiteDown()
{ID_Textaerea.doScroll("scrollbarPageDown");}
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick=" ScrolleSeiteRechts()">
    scrolle Seite rechts
</BUTTON>
<BR>
<BUTTON
    onclick=" ScrolleDown()"
    ondblclick=" ScrolleSeiteDown()"
>
    Texarea: Click = scrolle down Doppelklick = scrolle Seite down
</BUTTON>
<BR>
<BR>
<TEXTAREA ID="ID_Textaerea" >
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
</TEXTAREA>
</BODY>
```

.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht
.getAttribute()	DOM nicht geändert Wert eines per HTML erzeugten Attributes liefern
.getAttributeNode()	DOM nicht geändert Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar
.getBoundingClientRect()	DOM nicht geändert Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut):



	siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird



.select()	Objekt muss an sich schon renderbar sein Objekt selektieren z.B. Textbereich: wird hervorgehoben ControlRange: es wird Rechteck-Rahmen erzeugt nur unter Windows 32-Bit
Beispiel:	
	<pre> <SCRIPT LANGUAGE="JScript" FOR=document EVENT=onclick > var TextBereich = document.body.createTextRange(); TextBereich.moveToPoint(window.event.x, window.event.y); TextBereich.expand("word"); TextBereich.select(); </SCRIPT> </pre>
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.4.3.44. document.TextRange Objekt des Internet Explorer (Textbereich im Dokument)

Objekt des gesamten Plaintextes eines Objektes (Textbereich, Textrange)

basiert auf dem Objekt TextNode

Elternobjekte mit Textrange sind alle Objekte, die eine der nachfolgend beschriebenen Methoden besitzen
z.B. body Objekt
button Objekt
textarea Objekt
input text Objekt
selection Objekt (nur wenn ein Text selektiert wurde (mit oder ohne HTML))
können weitere HTML-Elemente enthalten, die ebenfalls Textbereiche besitzen können

Textbereich kann in Textteile zerlegt sein: Jeder Teil ist ein Element mit Plaintext.
hat seinen Bereichrahmen (siehe Objekt TextRange.TextRectangle und Collection TextRange.TextRectangle)

Erzeugung: in HTML

innerhalb der Tag-Begrenzer muss Text kodiert sein
z.B. leerer DIV hat keinen Textbereich
kann nachträglich einen Textbereich per Script erhalten

Beispiel: <HTML>
<BODY>
<H1>Hallo</H1>
<CENTER><H2> und willkommen ! </H2></CENTER>
</BODY>
</HTML>

Textrange ist erzeugt worden im BODY-Teil
und enthält nur Plain-Text also
"Hallo und willkommen !"
hat genau einen Zeiger auf die Textrange-Element

Textrange-Element ist z.B. das Wort "Hallo" des H1-Elementes im Container BODY

per Script:

.createTextRange() Textbereich erzeugen
Syntax:
[var Zeiger =] object.createTextRange()

Zeiger auf Range
null wenn TextRange nicht erzeugbar



Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var FeldAllerButtonElemente coll = document.all.tags("BUTTON");

    if (    (FeldAllerButtonElemente !=null )
        && (FeldAllerButtonElemente.length>0)
        )
    {
        var ZeigerAufRange = FeldAllerButtonElemente[0].createTextRange();
        ZeigerAufRange.text = "Clicked";
    }
</SCRIPT>

```

Beispiel 2:

```

function ErzeugeUndSelektiereTextRange()
{
    var TextBereich = document.body.createTextRange();
    TextBereich.findText("Testtext");
    TextBereich.select();
}

```

.createRange()

Zeiger auf einen Universal-Bereich erzeugen per document.selection Objekt des Dokumentes
Universal-Bereich kann enthalten:

Text (document.selection.textrange Collection
textrange Objekt)

oder Control-Element(e) (document.selection.controlRange Collection)

siehe auch Methoden .createControlRange() .createTextRange() und .createRangeCollection()
Eigenschaft .type

Syntax:

```
[ var Zeiger = ] document.selection.createRange()
```

.createRangeCollection()

document.selection.textrange Collection erzeugen per document.selection Objekt des Dokumentes
nur wenn der Browser multiple Selektion unterstützt, so mehr als 1 Feld-Element textrange Objekt
vorhanden bei Mehrfach-Selektion durch User

siehe auch textrange Objekt

Methoden .createRange() .createControlRange() und .createTextRange()

Syntax:

```
[ var Zeiger = ] = document.selection.createRangeCollection()
```

Eigenschaften:

.boundingHeight

Höhe des Rechteckes in Pixel um den Textbereich des TextRange Objektes

per textrange Objekt

.boundingLeft

Abstand linker Rand des Rechteckes in Pixel um den Textbereich zur linkem Rand des Container-Objektes,
in dem der Textbereich liegt

per textrange Objekt

.boundingTop
Objektes,

Abstand oberer Rand des Rechteckes in Pixel um den Textbereich zum oberen Rand des Container-

in dem der Textbereich liegt

per textrange Objekt

.boundingWidth

Breite des Rechteckes in Pixel um den Textbereich des TextRange Objektes

per textrange Objekt

.htmlText

HTML-Text im Textbereich
nicht den Plaintext-Anteil liefern
per textrange Objekt
nur unter Windows 32-Bit

.offsetLeft

X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem
des Elternobjektes (.offsetParent)

.offsetTop

Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem
des Elternobjektes (.offsetParent)

.text

Plain-Text im Textbereich
per textrange Objekt
Dokument muss komplett geladen sein
nur unter Windows 32-Bit

Methoden:

.collapse()

Textbereich-Zeichen-Zeiger auf Anfang oder Ende des Textbereiches setzten

Hinweis: Zeiger nur auf Plain-Text-Zeichen

Bsp.: <BODY><P>abc

Zeiger kann wandern von VOR a bis HINTER c

VOR a entspricht Start des Bereiches

mit Zeigerwert 0

HINTER c entspricht Ende des Bereiches

mit Zeigerwert 3

per textrange Objekt

nur unter Windows 32-Bit

.compareEndpoints()

Vergleich der Textbereich-Zeichen-Zeiger von 2 Textbereichen

Hinweis: Zeiger nur auf Plain-Text-Zeichen

Bsp.: <BODY><P>abc



Zeiger kann wandern von VOR a bis HINTER c
 VOR a entspricht Start des Bereiches
 mit Zeigerwert 0
 HINTER c entspricht Ende des Bereiches
 mit Zeigerwert 3

per textrange Objekt
 nur unter Windows 32-Bit
 .duplicate() Zeiger auf eine Duplikat-Instanz eines Textbereiches liefern
 per textrange Objekt
 nur unter Windows 32-Bit
 Prüfung eines Textbereiches auf Kopie eines anderen Textbereiches per Methode .inRange()
 bzw. .isEqual()
 .execCommand() Kommando ausführen z.B. im aktuellen Dokument
 in aktueller Selektion
 im aktuellen Bereich
 erst nach dem kompletten Laden des Dokumentes zulässig
 Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
 Control = Element zur Steuerung analog zum HTML-Element (Tag)
 Input-Control = Element mit Eingabeeigenschaft
 .expand() Plain-Text als Teil eines Textbereiches derart ausdehnen, dass er komplett die Dimension des Elternobjektes
 (Container-Objektes) einnimmt
 per textrange Objekt
 nur unter Windows 32-Bit
 .findText() Text im gesamten Textbereich suchen
 per textrange Objekt
 nur unter Windows 32-Bit
 für Nutzung einer Textmarke siehe Methoden .getBookmark() und .moveToBookmark()
 .getBookmark() Textmarke (Bookmark) im Textbereich setzen
 Textmarke kann mit Methode .moveToBookmark() anpositioniert werden
 per textrange Objekt
 nur unter Windows 32-Bit
 .getBoundingClientRect() Referenz auf TextRectangle-Objekt im Element holen
 .getClientRects() Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster
 Feld mit Index als Integer ab 0
 pro Eintrag ein Rectangle
 .inRange() prüfen ob 2 Textbereiche sich einschließen z.B. bei verschachtelten DIV's
 per textrange Objekt
 nur unter Windows 32-Bit
 .isEqual() Auf Identität zweier Textbereiche prüfen
 per textrange Objekt
 .move() Textbereich-Zeichen-Zeiger bewegen bezüglich aktueller Position im Textbereich
 per textrange Objekt
 nur unter Windows 32-Bit
 .moveEnd() Textbereich-Ende neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich
 per textrange Objekt
 nur unter Windows 32-Bit
 .moveStart() Textbereich-Anfang neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich
 per textrange Objekt
 nur unter Windows 32-Bit
 .moveToBookmark() Textbereich-Zeichen-Zeiger auf eine Textmarke (Bookmark) im Textbereich positionieren
 Textmarke wurde gesetzt per Methode .getBookmark()
 per textrange Objekt
 nur unter Windows 32-Bit
 .moveToElementText() Textbereich in ein Element/Objekt bewegen, das Text enthalten darf
 per textrange Objekt
 nur unter Windows 32-Bit
 .moveToPoint() Inhalt des Textbereiches um eine Pixelspanne verschieben (Offset) relativ zur linken oberen Fensterecke
 nach Verschiebung kann Textbereich leer sein
 per textrange Objekt
 nur unter Windows 32-Bit
 .parentElement() Zeiger auf das Elternelement (Container) des Textbereiches liefern
 Hinweis bei Elementverschachtelung:
 es wird das direkt um den Textbereich liegende Element referenziert
 per textrange Objekt
 nur unter Windows 32-Bit
 .pasteHTML() Textbereich-Inhalt durch Text ersetzen oder leeren Textbereich füllen
 Text kann auch HTML enthalten
 Tabelle nur mit kompletten HTML-Code einfügbar, also z.B. keine einzelne Zelle
 Achtung: Wenn HTML-Code im Text enthalten ist, muss das Elternobjekt
 auch diesen ansich unterstützen
 Bsp.: textArea erlaubt kein HTML-Code
 HTML-Code wird geparkt
 per textrange Objekt
 nur unter Windows 32-Bit



.queryCommandEnabled()	prüfen ob Kommando ausführbar ist
.queryCommandIndeterm()	prüfen ob Kommando-Status bestimmbar ist oder nicht
.queryCommandState()	Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
.queryCommandSupported()	prüfen ob Kommando im aktuellen Bereich unterstützt wird
.queryCommandValue()	Wert eines Kommandos liefern
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.select()	Objekt selektieren z.B. Textbereich: wird hervorgehoben ControlRange: es wird Rechteck-Rahmen erzeugt nur unter Windows 32-Bit
.setEndPoint()	Textbereichanfang bzw. -ende von 2 Textbereichen synchronisieren per textrange Objekt nur unter Windows 32-Bit

4.3.2.2.4.3.44.1. *document.TextRange.TextRectangle Collection des Internet Explorer*

ab IE 5.x

Feld der TextRectangle-Objekte im HTML-Element (Objekt) mit Plain-Text

TextRectangle-Objekt: Rahmen der Positionierung einer einzelnen Textzeile im Plain-Text
Positionierung bezüglich Koordinatensystem des HTML-Dokumentes

Beispiel: Ein DIV enthält Plain-Text:

Jede im Browser dargestellte Zeile (auch jedes einzelne
) besitzt einen eigenen Rechteck als
Rahmen der Positionierung innerhalb des DIV.
Damit lassen sich Textelemente per Rahmen innerhalb des Dokumentes positionieren.

Achtung: Nach einer Änderung der Fensterdimension (resize) muss die Collection neu erzeugt werden (ist zu programmieren !).

Syntax:

```
[ var FeldZeiger = ] zeiger_auf_textrange.getClientRects();
```

```
[ var FeldElementZeiger = ] FeldZeiger[Index];
```

Index: Integer und ab 0
muss in [] kodiert sein

Zugriff:

FeldZeiger.eigenschaft
FeldZeiger.methode()

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!
.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen
(analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.3.44.2. *document.TextRange.TextRectangle Objekt des Internet Explorer*

ab IE 5.x

TextRectangle-Objekt: Rahmen der Positionierung einer einzelnen Textzeile im Plain-Text (Positionierung bezüglich Koordinatensystem
des

HTML-Dokumentes)

instanziert als Element der Collection TextRange.TextRectangle

Beispiel: Ein DIV enthält Plain-Text:

Jede im Browser dargestellte Zeile (auch jedes einzelne
) besitzt einen eigenen Rechteck als
Rahmen der Positionierung innerhalb des DIV.
Damit lassen sich Textelemente per Rahmen innerhalb des Dokumentes positionieren.

Achtung: Nach einer Änderung der Fensterdimension (resize) muss die Collection neu erzeugt werden (ist zu programmieren !).

Syntax:

```
[ var Zeiger = ] zeiger_auf_textrectangle.getBndingClientRect();
```

zeiger_auf_textrectangle ist Element der Collection TextRange.TextRectangle

```
[ var Zeiger = ] zeiger_auf_textrectangle_collection[Index].getBndingClientRect();
```

Index Integer, ab 0
muss in [] kodiert

Beispiel:

```
<HEAD>
<SCRIPT>
var ZeigerAufTextRectangleCollection;
var Index =0;
```



```

function Anzeigen(Zeiger) // Zeiger auf den einzigsten DIV mit Text also mit Textbereich
{
    // aktuelle Collection der Rectangle von Div0 referenzieren:
    //      Das ist nötig nach resize des Fensters, da resize leider nicht automatisch erkannt wird

    ZeigerAufTextRectangleCollection = Zeiger.getClientRects();

    // Anzahl der Elemente ermitteln, also Anzahl der im Browser dargestellte Zeilen,
    //      wobei es egal ist, ob diese per <BR> erzeugt wurden oder nicht
    //      Hinweis: Zeilenlänge hängt auch von der Fensterbreite ab
    //      (automatischer Umbruch)
    // Jede Zeile besitzt ihr eigenes Rectangle
    AnzahlTextRectangle = ZeigerAufTextRectangleCollection.length;

    // aktuellen Index prüfen ob letzte Zeile bereits erreicht wurde
    if (Index > AnzahlTextRectangle -1) // Index ab 1, Anzahl ab 1
    {
        // es wurde die letzte Zeile erreicht

        // Div2 unsichtbar machen also entfärben
        // Hinweis: Div2 liegt auf dem Rectangle von Div0
        ID_Div2.style.display="none";

        // rücksetzen des Index, also mit erster Zeile weitermachen
        Index = 0;
    }

    // Rechteck der aktuellen Zeile ermitteln unter Beachtung eines eventuellen Scrollens
    var PosRechts = ZeigerAufTextRectangleCollection [Index].right + ID_Body.scrollLeft;
    var PosLinks = ZeigerAufTextRectangleCollection [Index].left + ID_Body.scrollLeft;
    var PosOben1 = ZeigerAufTextRectangleCollection [Index].top + ID_Body.scrollTop;

    // und Div1 auf die Zeile positionieren, also Zeile einfärben
    ID_Div1.style.top = PosOben1;
    ID_Div1.style.width = (PosRechts - PosLinks) - 5;
    ID_Div1.style.display = 'inline';

    // aktuelle Position des Rechteckes von DIV0 ermitteln unter Beachtung eines eventuellen
    // Scrollens
    PosRechts = Zeiger.getBoundingClientRect().right + ID_Body.scrollLeft;
    PosLinks = Zeiger.getBoundingClientRect().left + ID_Body.scrollLeft;
    var PosOben2 = Zeiger.getBoundingClientRect().top + ID_Body.scrollTop;

    // und Div2 überlagern positionieren
    ID_Div2.style.top = PosOben2;
    ID_Div2.style.width = (PosRechts - PosLinks) - 5;
    ID_Div2.style.height = PosOben1 - PosOben2;

    // aber nur einfärben, wenn mindestens 1 Zeile bereits eingefärbt wurde
    if (Index > 0){ ID_Div2.style.display = 'inline';}

    // Rectangle der nächsten Zeile einstellen
    Index++;
}
</SCRIPT>
</HEAD>
<BODY ID="ID_Body">
    <DIV ID="ID_Div0"
        onclick=" Anzeigen(this)"
    >
        klicke
        <BR>
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        123456789012345678901234567890
    </DIV>
    <DIV ID="ID_Div1"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:yellow; display:none"

```



```

>
</DIV>
<DIV ID="ID_Div2"
      STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:beige; display:none"
>
</DIV>
</BODY>

```

Zugriff:

Zeiger.eigenschaft

Eigenschaften:

sind les- und schreibbar

.bottom	untere Pixelposition des Rechteckes um ein Objekt
.left	linke Pixelposition des Rechteckes um ein Objekt
.right	rechte Pixelposition des Rechteckes um ein Objekt
.top	obere Pixelposition des Rechteckes um ein Objekt

Methoden:

keine

4.3.2.2.5. window.event Objekt

Instanz aller Ereignisse im Fenster

Ein Ereignis ist ein browserinternes Signal zu einer getätigten Aktion anhand des HTML-Elementes. Das Signal zeigt an, dass eine Aktion erfolgt ist bzw. noch andauert. Typisches Ereignis ist das Mausklicken des Users auf ein HTML-Element: Klickt der User, so wird die Aktion "Klick" per Ereignis "onclick" signalisiert, wenn das HTML-Element klickbar ist, also die Aktion "Klick" unterstützt.

Ein Ereignis kann von einem konkreten HTML-Element bzw. Objekt nur dann ausgelöst werden, wenn es Events überhaupt unterstützt. Nicht alle HTML-Elemente unterstützen Events. Die Event-Unterstützung von HTML-Elementen ist in der Scriptmaschine spezifisch zu jedem HTML-Element genau vordefiniert. Aus dieser Menge der vordefinierten Events zum HTML-Element kann der Programmierer schöpfen und Aktionen des Users zum HTML-Element zulassen oder unterbinden. Dabei gilt die Regel: Was nicht explizit zugelassen wurde, gilt als unterdrückt, falls es keine standardmäßige Zulassung gibt.

Ein HTML-Objekt kann natürlich nur dann Events auslösen, wenn es instanziiert ist. Beispiel für einen DIV: Wird dem DIV über die Eigenschaft .innerHTML nur ein Leerzeichen zugewiesen (zeiger_auf_div.innerHTML=' '; mit zeiger laut ID-Attribut-Wert), so gilt der DIV für die Eventerzeugung z.B. onmousedown als **nicht instanziiert**. Desweiteren darf der DIV per Style-Eigenschaft style.visibility mit Wert 'hidden' **nicht ausgeblendet** sein.

Pro Aktionsart existiert eine Eventart. Erfreulicherweise gibt es diverse Aktionen, die von vielen HTML-Elementen unterstützt werden, vorallem eben o.g. Useraktionen. Aber es gibt diverse Aktionen, von denen der User nichts mitbekommt. Die Eventbehandlung dient also nicht ausschliesslich der Aktionsfreudigkeit des Users und Programmierers, sondern ist ein Nebeneffekt in Form der interaktiven Webseite.

Viele Objekte können erst kommunizieren, wenn sie Events erzeugen bzw. ein Signal als Voraussetzung für eine Aktion bekommen. Dabei ist das Wort "Aktion" auch als "Verhalten" zu verstehen., also als Komponente zur Steuerung von HTML-Elementen während der Anzeige des HTML-Dokumentes im Browserfenster. Typisches Event ist das Signal "onload", das ausgelöst wird, wenn das HTML-Dokument komplett geparkt und falls nötig in das Browserfenster geladen **wurde**. Es gibt diverse Aktionen, die erst **nach** Auslösung von onload zulässig sind. Diese Aktionen betreffen auch die skriptgesteuerte Verwaltung des HTML-Dokumentes zu dessen Laufzeit.

Jedes instanziierte Objekt, das Events unterstützt, nutzt das Eventobjekt, mit dem die Verarbeitung aller unterstützten Events möglich ist (verschiedene Eventarten).

Per JScript kann ein Eventobjekt nur für das Objekt document erzeugt werden.

Aufgrund der Verschachtelung von HTML-Elementen müssen Events auch innerhalb der Hierarchie der HTML-Elemente **und** in der Vererbungsfolge verfügbar sein. Events können also durchgereicht werden. In Script wird üblicherweise die Punktnotation für Hierarchieebenen verwendet. Events können nur dann durchgereicht werden, wenn **beide betroffene Ebenen** diese Events unterstützen. Natürlich müssen die verschachtelten HTML-Elemente, welche Events durchreichen (und diese eventuell zuvor auswerten, wobei das Kind anders auf das Event reagieren kann als Eltern, die aber über das Ereignis immer informiert werden **sollten**), das Entstehen von Ereignissen auch überwachen, denn es muss nicht bekannt sein, wann und wo das Ereignis entsteht. Diese Überwachung kann in Verbindung mit Script genau gesteuert werden.

Standardgemäß gilt beim Internet Explorer: Kinder reichen Events zu den Eltern hoch und das bis zum BODY, also der obersten Dokumentenebene. Die Eltern haben standardgemäß immer das letzte Wort: Will ein Kind ein Event auslösen, so kann es das. Aber die Eltern entscheiden, ob das Event erhalten bleibt. Besonders gilt das für Body (document.body).

Beispiel: Wird per

```

document.body.onclick=OnClickHandler; // ohne () kodieren, damit nicht sofort ausgeführt wird
document.body.attachEvent('onclick',OnClickHandler);

```

ein Eventhandler instanziiert, so kann das Kind zwar **sein** onclick (eigener Handler muss im Kind implementiert sein) auslösen und entsprechende Reaktionen verursachen, aber die Eltern, also document.body, können innerhalb OnClickHandler() das Ereignis onclick sperren, so dass die Reaktionen des Kindes aufgehoben werden. Diese Reaktionen sind also nur solange wirksam, bis document.body das Event onclick erreicht hat. Fatal wird das bei Bildschirmausgaben des Kindes aufgrund onclick: Diese Ausgaben werden schlichtweg durch die Eltern aufgehoben (keine Anzeige mehr im Dokument).



Beispiel: Wird per

```
function OnContextMenuHandler()
// Unterbindung des Kontextmenüs
// Achtung: return false; unterdrückt nicht das Kontextmenü
{event.returnValue=false;}

document.body.oncontextmenu=OnContextMenuHandler;// ohne () kodieren, damit nicht sofort ausgeführt wird
document.body.attachEvent('oncontextmenu',OnContextMenuHandler);
```

ein Eventhandler zum Ereignis oncontextmenu implementiert, so bleibt der rechte Maustastendruck im gesamten Dokument und für alle Kinder wirkungslos, es sei denn, das Kind implementiert einen eigenen Handler für das Ereignis des Drückens der rechten Maustaste (z.B. per Event onmousedown).

Es existiert pro HTML-Element bzw. Objekt, das Events unterstützt, eine je nach Eventart vordefinierte Standardbehandlung. Aber Events können abweichend von der Standardbehandlung durch private Eventhandler in Script verarbeitet werden. Auch die Eventüberwachung kann in Script genau gesteuert werden: Es muss also nicht sein, dass ein Kind ein Event seinen Eltern mitteilt (darüber entscheidet der Programmierer).

Aufgrund dieser Tatsache sind Script-Methoden der Eventverwaltung nicht im Objekt event implementiert, sondern im Objekt, das das Event erzeugt. Die Implementierung ist objekt-spezifisch (siehe konkrete Objektbeschreibung) aber genormt. Es gibt also nicht viele jedoch wie immer browserspezifische Methoden.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizensierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
klick ins Fenster per aktivem Popup

Der Absturz ist "read"-Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).



Abänderungen wegen Browser-Inkompatibilität*Popupblocker-Fehler*

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren

oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster

einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer

anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen

Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

window.focus();

window.document.focus();

if(document.body!=null)

{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)

{ document.body.focus();}

}

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

popupzeiger.show(...);

Hinweis: Der Populfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus

.setActive() ist Teilmenge von .focus(): nur das aktiv setzen

funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.



Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•
http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}
 Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.



Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen: • 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen: • 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen: • http://msdn.microsoft.com/library/en-us/xmlsdk/hm/xml_concepts2_7ook.asp (http://msdn.microsoft.com/library/en-us/xmlsdk/hm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen: •

<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.



Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht



mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}
```

```
var X85=new Array();var X86=new Array();
```

```
X85[0]=window.open(...);
```

```
var X87='parent.Y_unload(0)'; X86[0]=new Function(",X87);
```

```
X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet, so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">  
unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild
```

```
<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12)">  
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12)">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein, innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

Instanz-Erzeugung in HTML:

Beispiele zur Kodierung von onXXX="..." im HTML-Tag des Objektes, wobei onXXX ein im Objekt implementiertes Event ist, z.B. onclick.

Übliche Kodierung:

```
<HEAD>  
<SCRIPT>
```



```

        function InputButton_Event_onclick_Handler()
        {alert("onclick erkannt");}
    </SCRIPT>
</HEAD>
<BODY>
    <INPUT  TYPE=button
           NAME="ID_Button"
           VALUE="Test"
           onclick="InputButton_Event_onclick_Handler()"
    >
</BODY>

```

Alternative 1:

```

<HEAD>
    <SCRIPT>
        function InputButton_Event_onclick_Handler()
        {alert("onclick erkannt");}
    </SCRIPT>
</HEAD>
<BODY>
    <INPUT  TYPE=button
           NAME="ID_Button"
           VALUE="Test"
    >
    <SCRIPT>
        // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
        // eine im HEAD deklarierte Funktion

        // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
        // geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
        ID_Button.onclick = InputButton_Event_onclick_Handler; //ohne ()
    </SCRIPT>
</BODY>

```

Alternative 2:

```

<BODY>
    <INPUT  TYPE=button
           NAME="ID_Button"
           VALUE="Test"
    >
    <SCRIPT>
        // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
        // eine im BODY deklarierte Funktion

        // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
        // geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
        function ID_Button.onclick()
        {alert("onclick erkannt");}
    </SCRIPT>
</BODY>

```

Alternative 3:

```

<HEAD>
    <SCRIPT>
        function InputButton_Event_onclick_Handler()
        {alert("onclick erkannt");}
    </SCRIPT>
</HEAD>
<BODY>
    <INPUT  TYPE=button
           NAME="ID_Button"
           VALUE="Test"
    >
    <SCRIPT>
        // anstelle der onclick-Kodierung im Tag erfolgt hier das Überschreiben des Standardeventhandlers durch
        // eine im BODY deklarierte Funktion, die eine Funktion im HEAD aktiviert

        // Überschreiben muss im BODY erfolgen, da der HEAD vor dem BODY-Teil
        // geparkt wird, also im HEAD das INPUT-Tag nicht bekannt ist
        function ID_Button.onclick()
        {InputButton_Event_onclick_Handler();}
    </SCRIPT>
</BODY>

```



Alternative 4 nur beim IE ab IE 4.x:

Das Script-Tag wird um die Attribute FOR und EVENT erweitert:

```
<SCRIPT FOR=objekt_bezeichner EVENT=event_bezeichner .....>
.....
</SCRIPT>
```

objekt_bezeichner	Standard-Objekt laut DOM z.B. window ID des instanziierten Objekt z.B. laut ID-Attribut Kodierung wahlweise mit oder ohne "" bzw. "
event_bezeichner	alle Events mit Präfix on z.B. onload siehe Objekt event Kodierung wahlweise mit oder ohne "" bzw. "

Ansonsten gilt das übliche zum Script-Tag, also auch die Lage im HEAD und/oder BODY.

Beispiel für Kodierung im BODY: Das Objekt muss während der Abarbeitung des HEAD nicht bereits bekannt sein.

```
<BODY>
  <INPUT TYPE=button
    NAME="ID_Button"
    VALUE="Test"
  >
  <SCRIPT FOR= ID_Button
    EVENT=onclick
  >
    alert("onclick erkannt");
  </SCRIPT>
</BODY>
```

Beispiel für Kodierung im HEAD: Das Objekt muss bereits während der Abarbeitung des HEAD bekannt sein.

```
<HTML>
<HEAD>
<SCRIPT FOR=window
  EVENT=onload
>
  var Filter0 = ID_Div.filters[0];
  Filter0.Apply();
  ID_Div.innerHTML= "<IMG SRC='test.jpg' WIDTH=300 HEIGHT=300>";
  Filter0.Play();
</SCRIPT>
</HEAD>
<BODY>
  <DIV ID= "ID_Div"
    STYLE= "position:absolute;width:300;height:300;top:20;left:20;
      filter:revealTrans(transition=12,duration=8)
    ">
    <BR>
    Dieser Text wird ueberblendet per Filter revealTrans (nur IE).
    Der Filter wird aktiv mit Laden des Dokumentes in das Fensters.
  </DIV>
</BODY>
</HTML>
```

Events bei sich überlagernden Objekten:

Beispiel: Sollte ein Image mit seiner Erzeugung ein anderes Image überlagern, so ist die Eventübergabe von und an das untere Image unterbrochen: Ereignisse onXXX kommen nicht mehr durch, solange das untere Bild nicht den **Fokus** erhält. Alternativ ist die Style-Eigenschaft z-index zu kodieren und dann der z-index auf den obersten zu setzen.

4.3.2.2.5.1. Eventarten Internet Explorer und Netscape

4.3.2.2.5.1.1. Eventarten der HTML-Tags (Auswahl)

onabort	meldet Abbruch des Ladens	IMG
onblur	meldet Verlassen/Deaktivieren eines HTML-Elementes	BODY BUTTON CHECKBOX FILEUPLOAD FRAMESET Frame-Objekt



		PASSWORD RADIO RESET SELECT SUBMIT TEXT TEXTAREA Window-Objekt
onchange	meldet Veränderung eines HTML-Elementes	FILEUPLOAD SELECT TEXT TEXTAREA
onclick	meldet Mausklick auf HTML-Element	BUTTON CHECKBOX Link per HREF es wird erst gemeldet, dann das Link angesprungen Handler muss return true; bzw. return false; besitzen true --> Aktion des HTML-Elementes wird zugelassen false --> Aktion des HTML-Elements wird nicht zugelassen
		RADIO RESET SUBMIT
onerror	meldet Fehler bei Ausführung der Aktion zum HTML-Element	IMG (Fehler bei Bildladen) Window-Objekt (Fehler bei Ausführung Javascript)
onfocus	meldet Aktivierung des HTML-Elementes	BODY BUTTON CHECKBOX FILEUPLOAD FRAMESET Frame-Objekt PASSWORD RADIO RESET SELECT TEXT TEXTAREA Window-Objekt
onload	meldet komplettes Laden des HTML-Elementes	BODY FRAME FRAMESET IMG Window-Objekt
onmouseout	meldet, dass Mauszeiger nicht auf HTML-Element steht	A AREA Link per HREF
onmouseover	meldet, dass Mauszeiger sich über Element bewegt	A AREA Link per HREF
onreset	meldet gedrückten Reset-Button	FORM Handler muss return true; oder return false; besitzen: true --> Reset wird ausgeführt false --> Reset wird nicht ausgeführt
onselect	meldet, dass der Benutzer einen Textausschnitt markiert hat	



TEXT
TEXTAREA

onsubmit meldet gedrückten Submit-Button
FORM
Handler muss return true; oder return false; besitzen:
true --> Submit wird ausgeführt
false --> Submit wird nicht ausgeführt
bei E-Mail erfolgt automatische Anfrage,
ob wirklich gesendet werden soll

onunload meldet Verlassen des HTML-Elementes
BODY
FRAME
FRAMESET
Window-Objekt

4.3.2.2.5.1.2. Eventarten des IE und NS (Auswahl)

onabort Ladevorgang eines Bildes bricht ab z.B. wegen Useraktion Stop
Tag
Objekt image
IE ab 4.x, NS ab 3.x

onafterprint Druck eines Fensters oder Frames: Ereignis, das direkt nach dem Druckende
ausgelöst wird
Tag <BODY>, <FRAMESET>
Objekt window
nur IE ab 5.x

onbeforecopy markierten Text in die Zwischenablage kopieren:
Ereignis, das direkt vor der Kopieraktion ausgelöst wird
Tag alle Tags mit markierbarem Text
z.B. <A>, <DIV>, , <TEXTAREA>
Objekt alle Objekte mit markierbaren Text
nur IE ab 5.x

onbeforecut markierten Text ausschneiden und in die Zwischenablage einfügen:
Ereignis, das direkt vor dem Ausschneiden ausgelöst wird
verlangt im Händler event.returnValue = false;
für Erzeugung der Kontextmenü-Eintrages zur Ausschneide-Operation
(standardmäßig ist Ausschneiden nicht möglich)
Tag alle Tags mit markierbarem Text
z.B. <A>, <DIV>, , <TEXTAREA>
Objekt alle Objekte mit markierbaren Text
nur IE ab 5.x

onbeforepaste Text aus Zwischenablage einfügen:
Ereignis, das direkt vor dem Einfügen ausgelöst wird
verlangt im Händler event.returnValue = false;
für Erzeugung der Kontextmenü-Eintrages zur Einfüge-Operation
(standardmäßig ist Einfügen nicht möglich)
Einfüge-Operation im Kontextmenü ermöglichen:
Handler muss return false; bzw. returnValue=false; liefern
Tag alle Tags mit markierbarem Text
z.B. <A>, <DIV>, , <TEXTAREA>
Objekt alle Objekte mit markierbaren Text
nur IE ab 5.x

onbeforeprint Druck eines Fensters oder Frames: Ereignis, das direkt vor dem Druckstart
ausgelöst wird
Tag <BODY>, <FRAMESET>
Objekt window
nur IE ab 5.x

onbeforeunload Verlassen der HTML-Seite: Ereignis, das direkt vor dem Verlassen ausgelöst wird
wenn im Handler event.returnValue = 'freier_melde_text'; so wird automatisch
vor dem Verlassen ein Anfragefenster geöffnet z.B. Anfrage, ob die
HTML-Seite wirklich verlassen werden soll
Tag <BODY>, <FRAMESET>
Objekt window
nur IE ab 4.x

onblur Ereignis, das direkt vor dem De-Fokussieren ausgelöst wird
Tag <FRAME>, <INPUT>, <TEXTAREA>
IE ab 4.x, NS ab 3.x

onchange Ereignis ausgelöst, wenn Inhalt von Eingabefeld
Textarea
Auswahlliste
Radiobox (nur IE)
Checkbox (nur IE)
durch User verändert wurde
und bei Eingabefeld, Textarea und Auswahlliste



	diese de-fokussiert wurden (dagegen bei Radiobox und Checkbox sofort Ereignisauslösung)
	Handler hat innerhalb <OPTION> eines <SELECT> keine Wirkung, also nur innerhalb <SELECT> kodieren
	Tag <INPUT>, <SELECT>, <TEXTAREA>
	IE ab 4.x, NS ab 3.x
onclick	Ereignis ausgelöst, wenn Maus das Objekt angeklickt hat: Taste drücken und loslassen im Handler return false; kodieren für Links, Formularelemente und bei IE auch DIV wenn eine Aktion nicht erwünscht ist Bsp.: bei Link wird dieser nicht ausgeführt, obwohl angeklickt
	Tag <A>, <BODY>, <INPUT>, nur bei IE: <DIV>
	Objekt document, button, submit, reset, checkbox
	IE ab 4.x, NS ab 3.x
oncontextmenu	Ereignis ausgelöst direkt vor Aufruf des Kontextmenüs mit der rechte Maustaste Unterdrückung, wenn der Eventhandler event.returnValue=false; enthält (mit return false; erfolgt keine Unterdrückung) gilt standardgemäß HTML-seitenweit, also für alle Elemente der Seite, da das Ereignis standardgemäß bis nach oben weitergereicht wird
	Tag fast alle
	Objekt analog zu Tag z.B. document, link
	nur IE ab 4.x
	Hinweis für NS: Kontextmenü sperren per Ereignishandler für mousedown der rechten Maustaste: nur für Object document muss return false; liefern
oncopy	markierten Text in die Zwischenablage kopieren: Ereignis, das direkt nach der Kopieraktion ausgelöst wird
	Tag alle die Text definieren, <A>
	Objekt analog zu Tag z.B. link
	nur IE ab 5.x
oncut	markierten Text ausschneiden und in die Zwischenablage einfügen: Ereignis, das direkt mit dem Ausschneiden ausgelöst wird verlangt im Handler event.returnValue = false; für Aktivierung des Kontextmenü-Eintrages zum Ausschneiden (standardmäßig ist Ausschneiden nicht möglich)
	Tag alle die Text definieren z.B. <A>
	Objekt analog zu Tag z.B. link
	nur IE ab 5.x
ondblclick	Ereignis ausgelöst, wenn Doppelklick auf Objekt erfolgt 1 Klick = Maustaste drücken und loslassen bei IE: nur durch linke Maustaste bei NS: durch linke bzw. rechte Maustaste
	Tag fast alle
	Objekt link
	bei IE: nur durch linke Maustaste bei NS: durch linke bzw. rechte Maustaste
	IE ab 4.x, NS ab 4.x
ondrag	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, immer wenn HTML-Element verschoben wird, also permanent solange gezogen wird nur verwenden für Erfassung der aktuellen Element-Position Quell- und Ziel-Element müssen gleicher Art sein Eventhandler für Quellobjekt kodieren: nur auf Ereignis reagieren, solange der Bereich des Zielobjektes noch nicht erreicht wurde private Boolean-Variable verwenden: if (false) ... reaktion per dragenter auf true setzen per dragleave auf false initialisieren
	Tag alle Tags, in denen Text markierbar sind: <DIV>, ,
	Objekt analog zu Tag
	nur IE ab 5.x
ondragdrop	NS ermöglicht Drag&Drop von Dateien und Verknüpfungen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn eine Datei oder Verknüpfung verschoben wurde
	Tag <BODY>



	Objekt window nur NS ab 4.x
ondragend	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn HTML-Element verschoben, eventuell abgelegt und somit Drag & Drop nun beendet werden kann aktivieren des Ablegen per dragover Quell- und Ziel-Element müssen gleicher Art sein Eventhandler für Quellobjekt kodieren Tag alle Tags, in denen Text markierbar sind: <DIV>, , Objekt analog zu Tag nur IE ab 5.x
ondragenter	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn der Bereich des Ziel-Elementes das 1. Mal betreten wurde Eventhandler für Zielobjekt kodieren: muss die private Boolean-Variable auf true setzen per drag ausgewertet (dort auf false prüfen) per dragleave auf false initialisieren Quell- und Ziel-Element müssen gleicher Art sein Tag alle Tags, in denen Text markierbar sind: <DIV>, , Objekt document und analog zu Tag nur IE ab 5.x
ondragleave	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn während des Ziehens die Maustaste losgelassen wurde, wobei somit Drag & Drop zugleich abgebrochen wurde Eventhandler für Quellobjekt kodieren: muss die private Boolean-Variable auf false initialisieren per drag ausgewertet (dort auf false prüfen) per dragenter auf true gesetzt Quell- und Ziel-Element müssen gleicher Art sein Tag alle Tags, in denen Text markierbar sind: <DIV>, , Objekt document und analog zu Tag nur IE ab 5.x
ondragover	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn der Bereich des Ziel-Elementes ab 2. Mal betreten wurde also permanent fortlaufend, solange Maus im Bereich des Zieles ist und nicht abgeleget wurde Eventhandler für Zielobjekt kodieren: muss event.returnValue=false; enthalten, wenn auch Ablegen im Zielobjekt gewollt ist, also der Einfüge-Cursor mit Pfeil und Pluszeichen angezeigt werden soll (standardgemäß ist keine Ablage möglich, also Cursor mit dem durchgestrichenen Kreis sichtbar !) Quell- und Ziel-Element müssen gleicher Art sein Tag alle Tags, in denen Text markierbar sind: <DIV>, , Objekt document und analog zu Tag nur IE ab 5.x
ondragstart	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn Maus auf Quellobjekt dauergedrückt ist, also die Verschiebung damit beginnen kann Eventhandler für Quellobjekt kodieren Quell- und Ziel-Element müssen gleicher Art sein Tag alle Tags, in denen Text markierbar sind: <DIV>, , Objekt document und analog zu Tag nur IE ab 5.x
ondrop	IE ermöglicht Drag&Drop von HTML-Elementen auch über Fenster per Maus: Click auf Element und Maus gedrückt lassen, dann Element ziehen an gewünschte Position, dann Maus loslassen Ereignis ausgelöst, wenn Quellobjekt auf dem Bereich des Zielobjektes



	<p>abgelegt wird, falls es per dragend zugelassen wird</p> <p>Eventhandler für Zielobjekt kodieren</p> <p>Quell- und Ziel-Element müssen gleicher Art sein</p> <p>Tag alle Tags, in denen Text markierbar sind: <DIV>, , </p> <p>Objekt document und analog zu Tag</p> <p>nur IE ab 5.x</p>
onerror	<p>Ereignis ausgelöst, wenn</p> <ul style="list-style-type: none"> während Ladens eines HTML-Dokumentes ein Syntaxfehler auftritt eines Bildes ein Fehler auftritt Abarbeitung von Scripten ein Runtime-Error auftritt <p>sämtliche Fehlermeldungen unterbinden per</p> <pre>var RetteOnErrorHandler = window.onerror; window.onerror = null;</pre> <p>Eventhandler muss wie folgt kodiert werden:</p> <pre>function freier_name_fuer_onerror_behandlung (error_erklaerung_string, url_des_html_dokumentes_als_string, zeilen_nr_des_errors_im_html_dokument) { return true; // nur wenn true geliefert, dann // wird die Browsereigenen // onerror-Behandlung // unterdrückt }</pre> <p>pro Fehler ein Aufruf des Eventhandlers</p> <p>--> Folge von Fehlern, also Folge von Aufrufen</p> <p>Die Script-Debugger-Aufforderung ist nur in den Eigenschaften des Browsers abschaltbar</p>
onfilterchange	<p>Tag </p> <p>Objekt window, img</p> <p>IE ab 4.x und NS ab 3.x</p> <p>IE unterstützt Filter-Effekte per CSS: STYLE=".....filter:....."</p> <p>z.B. filter: revealTrans(Transition = 5, Duration = 1.0)</p> <p>Ereignis ausgelöst mit Ablaufende des Filtereffektes</p> <p>Tag <DIV>, , <INPUT>, , <TABLE>, <TD>, <TEXTAREA>, <TH>, <TR></p> <p>Objekt image</p> <p>nur IE ab 4.x</p>
onfocus	<p>Ereignis ausgelöst, wenn HTML-Element fokussiert wird, also das aktuelle wird</p> <p>in Handler keine alert(...)-Anweisung kodieren !!!</p> <p>Tag IE und NS: <BODY>, <DIV>, <INPUT>, <FRAMESET>, <TEXTAREA></p> <p>nur IE: <APPLET>, , , <FORM>, <IFRAME>, , <PRE>, <TABLE>, <TD>, <TH>, <TR>, </p> <p>Objekt window bei IE zusätzlich image</p> <p>IE ab 4.x, NS ab 3.x</p>
onhelp	<p>Ereignis ausgelöst, wenn im Browserfenster die F1-Taste gedrückt wurde</p> <p>Handler muss return true; liefern, wenn eine Reaktion auf die F1-Taste auch wirksam werden soll</p> <p>Tag <BODY>, <FRAMESET></p> <p>Objekt window</p> <p>nur IE ab 4.x</p>
onkeydown	<p>Ereignis ausgelöst</p> <ul style="list-style-type: none"> beim IE nicht für alle Tastenarten beim NS für alle Tastenarten <p>liefert ASCII-Code der Taste nach Eigenschaft</p> <ul style="list-style-type: none"> bei IE .keyCode bei NS .which <p>nach String umwandeln per string_name=fromCharCode(..)</p> <p>NS einmalig, also genau wenn Taste gedrückt wird</p> <p>IE erste Mal, wenn Taste gedrückt</p> <p>periodisch, wenn Dauerdruck der Taste</p> <p>--> siehe Eigenschaft .repeat</p> <p>Besonderheit bei Kombination Steuertaste und andere Taste</p> <p>z.B. Shift-Taste und B-Taste zugleich für Großbuchstabe B</p> <p>Ereignis wird pro Tastendruck aufgerufen, also</p> <ol style="list-style-type: none"> 1. Mal für Shift-Taste 2. Mal für B-Taste <p>ab IE 5.x und NS 4.x:</p> <p>keydown-Ereignis ruft automatisch das keypress-Ereignis auf, aber wenn keydown-Handler return false; bzw. beim IE alternativ event.returnValue=false; liefert, so wird ein kodierter keypress-Handler nicht aktiviert !</p> <p>Ereignisprogrammierung unterscheidet sich nicht nur zwischen den Browsern</p>



	sondern auch innerhalb deren Versionen !!!
Tag	bei IE <BODY>, <DIV>, <FORM>, <INPUT>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TR>, alle Text-Tags wie z.B. ,
	bei NS <A>, , <TEXTAREA>
Objekt	document
ab IE 4.x bzw. 5.x und NS 4.x	
onkeypress	wird unmittelbar nach dem keydown-Ereignis ausgelöst allerdings nur für Tastenarten ! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ ' " ` ~ 0 bis 9 a bis z beim IE: egal ob Gross oder Klein, da immer der ASCII-Code des Grossbuchstaben geliefert wird Enter Leertaste und dient dem Erkennen von Tastaturkombinationen, die nicht vom Ereignis keydown abgefangen werden der Aufruf des keypress-Handlers ist nur möglich, wenn der keydown-Handler return true; bzw. beim IE alternativ event.returnValue=true; liefert
	Ereignisprogrammierung unterscheidet sich nicht nur zwischen den Browsern sondern auch innerhalb deren Versionen !!!
Tag	bei IE <BODY>, <DIV>, <FORM>, <INPUT>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TR>, alle Text-Tags wie z.B. ,
	bei NS <A>, , <TEXTAREA>
Objekt	document
ab IE 4.x bzw. 5.x und NS 4.x	
onkeyup	Ereignis ausgelöst, wenn gedrückte Taste jeder Art losgelassen wird wird automatisch nach Ereignis keypress ausgelöst Ereignis keypress automatisch nach Ereignis keydown ausgelöst bei NS: besitzt vordefiniertes Schlüsselwort KEYUP für captureEvents also document.captureEvent(Event.KEYUP); Ereignisprogrammierung unterscheidet sich nicht nur zwischen den Browsern sondern auch innerhalb deren Versionen !!!
Tag	bei IE <BODY>, <DIV>, <FORM>, <INPUT>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TR>, alle Text-Tags wie z.B. ,
	bei NS <A>, , <TEXTAREA>
Objekt	document
ab IE 4.x bzw. 5.x und NS 4.x	
onload	Ereignis ausgelöst, wenn vollständig geladen wurde HTML-Dokument mit all seinen Elementen jeder Art Framset (innerhalb <FRAMESET> kodieren) Bild DIV bei NS zusätzlich Layer bei IE zusätzlich Applet, Script, Link, IFRAME zu animiertes Bild (Gif-Datei): bei NS: Ereignis load bei jedem Bildwechsel der Animation ausgelöst es kann also für jedes Bild das Ereignis behandelt werden bei IE: Ereignis load bei jeder Animationswiederholung ausgelöst es kann nur pro komplette Animation das Ereignis behandelt werden
Tag	<BODY>, <DIV>, <FRAMESET>, bei IE zusätzlich <A>, <APPLET>, <IFRAME>, <SCRIPT>
Objekt	window, img
ab IE 4.x und NS 3.x	
onlosecapture	Ereignis ausgelöst, wenn releaseCapture() für mousemove, mouseover und mouseout ausgeführt wurde der User mit der Maus das Browserfenster verlässt
Tag	<A>, <APPLET>, <BODY>, <DIV>, <FORM>, <INPUT>, <P>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TR>
nur IE 5.x	
onmousedown	Ereignis ausgelöst bei IE mit Drücken irgendeiner Maustaste siehe Event-Eigenschaft .button bei NS mit Drücken der linken oder rechten Maustaste (mittlere nicht !) siehe Event-Eigenschaft .which auch in Verbindung mit Umschalt (Shift) und Alt



	<p>bei IE: return false; bzw. returnValue=false; unterbinden nicht die Ausführung des Eventhandlers also Unterbindung z.B. in click oder contextmenu kodieren</p> <p>bei verschachtelt Objekten wird immer vom innen nach außen (unten nach oben) das Ereignis geliefert, solange bis das Event behandelt wurde kann per event.cancelBubble das Weiterreichen nach oben verhindert werden (in allen Ebene, die oberhalb derjenigen Ebene liegt, in der cancelBubble aktiviert wurde)</p> <p>bei NS: return false; unterbindet immer die Ausführung des Eventhandlers return false bei rechter Maustaste auf Object document wird das Kontextmenü gesperrt</p> <p>Tag <A>, <DIV> bei NS zusätzlich <INPUT TYPE="button"> bei IE zusätzlich <APPLET>, <AREA>, <BODY>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.</p>
onmouseenter	<p>Objekt document ab IE 4.x und NS 4.x Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt Tag unbekannt nur für IE, wahrscheinlich ab 5.02 Empfehlung: mouseover verwenden</p>
onmouseleave	<p>Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt Tag unbekannt nur für IE, wahrscheinlich ab 5.02 Empfehlung: mouseout verwenden</p>
onmousemove	<p>Ereignis bei jeder Mausbewegung ausgelöst Auswertung für den Bereich eines Objektes durch dessen Eventhandler Tag bei NS unbekannt bei IE <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.</p>
onmouseout	<p>Objekt document ab IE 4.x und NS 4.x Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt Tag <A>, <DIV>, bei IE zusätzlich <APPLET>, <AREA>, <BODY>, <FORM>, <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.</p>
onmouseover	<p>Objekt document ab IE 4.x und NS 3.x Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt Tag <A>, <DIV>, bei IE zusätzlich <APPLET>, <AREA>, <BODY>, <FORM>, <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.</p>
onmouseup	<p>Objekt document ab IE 4.x und NS 3.x Ereignis ausgelöst bei IE mit Loslassen irgendeiner Maustaste siehe Event-Eigenschaft .button bei NS mit Loslassen der linken Maustaste (rechte und mittlere nicht !) siehe Event-Eigenschaft .which bei IE: return false; bzw. returnValue=false; unterbinden nicht die Ausführung des Eventhandlers also Unterbindung z.B. in click oder contextmenu kodieren bei verschachtelt Objekten wird immer vom innen nach außen (unten nach oben) das Ereignis geliefert, solange bis das Event behandelt wurde kann per event.cancelBubble das Weiterreichen nach oben verhindert werden (in allen Ebene, die oberhalb derjenigen Ebene liegt, in der cancelBubble aktiviert wurde)</p> <p>bei NS: return false; unterbindet immer die Ausführung des Eventhandlers Tag <A>, <DIV> bei NS zusätzlich <INPUT TYPE="button"> bei IE zusätzlich <APPLET>, <AREA>, <BODY>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.</p> <p>Objekt document</p>



onmove	<p>ab IE 4.x und NS 4.x Ereignis ausgelöst solange ein Fenster verschoben wird Tag keine nur für Objekt window zulässig nur NS ab 4.x</p>
onpaste	<p>Text aus Zwischenablage einfügen: Ereignis, das direkt mit dem Einfügen ausgelöst wird verlangt im Handler event.returnValue = false; für Aktivierung des Kontextmenü-Eintrages zum Einfügen (standardmäßig ist Einfügen nicht möglich) Tag alle Tags mit markierbarem Text z.B. <A>, <DIV>, , <TEXTAREA> Objekt link nur IE ab 5.x</p>
onpropertychange	<p>Ereignis, das mit der Veränderung des HTML-Elementes eintritt Abfangen des Ereignisses von verschachtelten Elementen: jedes Elementes für sich: dort je den Eventhandler per onpropertychange kodieren das übergeordnete für alle seine untergeordneten: nur im übergeordneten Element den Handler kodieren Objekt feststellbar durch event.srcElement.name laut NAME-Attribut im HTML-Element bzw. event.srcElement.id laut ID-Attribut im HTML-Element Eigenschaft feststellbar durch event.propertyName Tag <A>, <APPLET>, <AREA>, , <BODY>, <CODE>, <DIV>, <DL>, , <FORM>, <I>, , <INPUT>, , <MAP>, , <OPTION>, <P>, <PRE>, <SCRIPT>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TH>, <TR>, nur IE ab 5.x</p>
onreset	<p>Ereignis VOR dem Rücksetzen eines Formulars ausgelöst Rücksetzen per Reset-Button bzw. Neuladen des Dokumentes dient zum Abfangen der Resetaktion des Users und eventueller Unterbindung des Resets z.B. wegen falscher Ausfüllung des Formulars Reset wird nicht ausgelöst, wenn Handler return false; liefert Tag <FORM> Objekt form ab IE 4.x und NS 3.x</p>
onresize	<p>Ereignis mit ausgelöst bei Größenänderung des Fensters per Maus: Maus auf Fensterrahmen es erscheint das <> Symbol linke Maustaste drücken, gedrückt lassen und ziehen linke Maus loslassen bei IE: mit jeder Veränderung der Größen durch ziehen, also jede Veränderung einzeln behandelbar bei NS: genau für die letzte Größenänderung des Fensters, also mit Loslassen der linken Mausänderung Ereignis muss beim IE abgefangen werden per logischer_window_name.attachEvent('onresize', event_handler); Tag bei NS: keins, da nur für window-Objekt zulässig bei IE: neben dem window-Objekt auch <A>, , <CODE>, <DIV>, <FORM>, <FRAME>, <HR>, <I>, , <INPUT>, , <P>, <PRE>, <SELECT>, , <STYLE>, <TABLE>, <TEXTAREA>, <U>, , Stylesheet Objekt window ab IE 4.x und NS 3.x</p>
onresizeend	<p>Ereignis ausgelöst, wenn Größe einer Markierung verändert wurde Tag <A>, , <BODY>, <CODE>, <DIV>, <FORM>, <FRAME>, <HR>, <I>, , <INPUT>, <P>, <PRE>, <SELECT>, , <TABLE>, <TEXTAREA>, <U> Objekt window, document nur IE ab 5.5</p>
onresizestart	<p>Ereignis ausgelöst mit erster Veränderung der Größe einer Markierung Tag <A>, , <BODY>, <CODE>, <DIV>, <FORM>, <FRAME>, <HR>, <I>, , <INPUT>, <P>, <PRE>, <SELECT>, , <TABLE>, <TEXTAREA>, <U> Objekt window, document nur IE ab 5.5</p>
onscroll	<p>Ereignis ausgelöst, wenn HTML-Element gescrollt wird z.B. HTML-Dokument --> Handler innerhalb <BODY> kodieren Tag <APPLET>, <BODY>, <DIV>, <EMBED>, <MAP>, <TABLE>, <TEXTAREA> nur IE ab 5.5</p>



onselect	Ereignis ausgelöst wenn sich Textmarkierung ändert Tag <code><INPUT TYPE="text"></code> , <code><TEXTAREA></code> bei IE zusätzlich <code><BODY></code> ab IE 4.x und NS 3.x
onstop	Ereignis ausgelöst, wenn Stop-Button des Browsers gedrückt wurde Seite verlassen wird auch durch Browser-Buttons Objekt <code>document</code> nur IE ab 5.x
onsubmit	Ereignis VOR dem Abschicken eines Formulars ausgelöst Abschicken per Submit-Button dient zum Abfangen der Resetaktion des Users und eventueller Unterbindung des Submits z.B. wegen falscher Ausfüllung des Formulars Submit wird nicht ausgelöst, wenn Handler return false; liefert Tag <code><FORM></code> Objekt <code>form</code> ab IE 4.x und NS 3.x
onunload	Ereignis ausgelöst, wenn das HTML-Dokument verlassen wird durch Schliessen Browserfenster Wechsel zu anderer Seite auch per Browser-Button <code>window.document.open()</code> <code>window.document.write()</code> Tag <code><BODY></code> , <code><FRAMESET></code> Objekt <code>window</code> ab IE 4.x und NS 3.x

4.3.2.2.5.2. **Eigenschaften im IE und NS (Auswahl)**

Die Eigenschaften gelten nur für die zugehörige Eventart:

z.B. gilt die Eigenschaft

`.button` NUR für ein Mausereignis

`.altKey` NUR für ein Tastaturereignis

oder sind teilweise auf genau eine Ereignisart zugeschnitten

z.B. `.toElement` für `onmouseover`

unterscheiden sich zwischen Internet Explorer und Netscape:

die wenigsten gelten für beide Browser

<code>.altKey</code>	true wenn ALT-Taste gedrückt, sonst false (egal ob linke oder rechte) nur IE ab 4.x
<code>.altLeft</code>	true wenn linke ALT-Taste gedrückt, sonst false nur IE ab 5.5 Hinweis: rechte Alt-Taste (AltGr) ermitteln aus (<code>altKey && (!altLeft)</code>)
<code>.button</code>	0 keine Maustaste gedrückt 1 linke Maustaste gedrückt 2 rechte Maustaste gedrückt 4 mittlere Maustaste gedrückt Kombination aus 1 bis 4 für Maustastenkombination z.B. 3 = linke UND rechte Maustaste gedrückt ($1 + 2 = 3$) 7 = alle Maustasten gedrückt ($1 + 2 + 3 + 4 = 7$)
<code>.cancelBubble</code>	nur IE ab 4.x true, so Ereignis an das übergeordnete Event weiterleiten, sonst false nur IE ab 4.x
<code>.clientX</code>	horizontale Pixel-Position der Maus gegen über linke obere Ecke (0,0) des Anzeigebereiches im Browserfenster nur IE ab 4.x Hinweis: horizontale Pixel-Position der Maus gegenüber der linken oberen Ecke des HTML-Dokumentes ergibt sich aus <code>event.clientX + document.body.scrollLeft</code>
<code>.clientY</code>	vertikale Pixel-Position der Maus gegen über linke obere Ecke (0,0) des Anzeigebereiches im Browserfenster nur IE ab 4.x Hinweis: vertikale Pixel-Position der Maus gegenüber der linken oberen Ecke des HTML-Dokumentes ergibt sich aus <code>event.clientY + document.body.scrollTop</code>
<code>.ctrlKey</code>	true, so Strg-Taste gedrückt (egal ob linke oder rechte) nur IE ab 4.x
<code>.ctrlLeft</code>	true wenn linke Strg-Taste gedrückt, sonst false nur IE ab 5.5 Hinweis: rechte Strg-Taste ermitteln aus (<code>ctrlKey && (!ctrlLeft)</code>)
<code>.data[]</code>	Feld der per Urls aller per Drag & Drop auf das HTML-Dokument abgelegten Objekte nur NS ab 4.x mit signiertem Script
<code>.dataTransfer</code>	selbst Objekt für Cut & Paste-Operationen für Texte oder Urls nur IE ab 5.x mit folgenden Methoden <code>dataTransfer.setData()</code>



	<code>.dataTransfer.getData()</code> <code>.dataTransfer.clearData()</code>
<code>.fromElement</code>	enthält Zeiger desjenigen Objektes, das das Ereignis <code>onmouseout</code> hat nur IE ab 4.x siehe auch <code>.toElement</code>
<code>.height</code>	Höhe in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde nur NS ab 4.x
<code>.keyCode</code>	ASCII-Code der gedrückten Taste wenn 0 so keine Taste gedrückt Umwandlung in String per <code>String.fromCharCode(TastencodeASCII)</code> nur IE ab 4.x
<code>.layerX</code>	horizontale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis darf kein <code>resize</code> -Event sein neue Breite in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Breite verändert wurde, nur <code>resize</code> -Event
<code>.layerY</code>	nur NS ab 4.x vertikale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis darf kein <code>resize</code> -Event sein neue Höhe in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Höhe verändert wurde, nur <code>resize</code> -Event
<code>.modifiers</code>	nur NS ab 4.x Bitmaske für Zustand der Steuertasten Shift, Alt und Strg dient zur Ermittlung der Steuertaste per vordefinierter Maske durch bitweise UND-Verknüpfung mit der Bitmaske ergibt die Verknüpfung 1, also true, so Steuertaste gedrückt worden vordefinierte Maske für Alt-Taste <code>"Event.ALT_MASK"</code> Strg-Taste <code>"Event.CTRL_MASK"</code> Shift-Taste <code>"Event.SHIFT_MASK"</code> Bsp: <code>return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0);</code> <code>// nicht logisches UND (&&) sondern bitweises UND also &</code>
<code>.offsetX</code>	nur NS ab 4.x horizontale Pixel-Position der Maus im HTML-Element z.B. <code><DIV></code> , in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberen Ecke (0,0)
<code>.offsetY</code>	nur IE ab 4.x vertikale Pixel-Position der Maus im HTML-Element z.B. <code><DIV></code> , in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberen Ecke (0,0)
<code>.pageX</code>	nur IE ab 4.x horizontale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
<code>.pageY</code>	nur NS ab 4.x vertikale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
<code>.propertyName</code>	nur NS ab 4.x enthält Bezeichner der geänderten Eigenschaft nur Verwenden als Parameter des <code>onpropertychange</code> -Handler
<code>.repeat</code>	nur IE ab 5.x true, so Taste im Dauerdruck, sonst false nur IE ab 5.x
<code>.returnValue</code>	enthält den Booleen-Rückkercode des Eventhandlers <code>event.returnValue=true;</code> ist identisch <code>return true;</code> <code>event.returnValue=false;</code> ist identisch <code>return false;</code> Eventhändler muss liefern true, um eine Aktion aufgrund des Events zuzulassen false, um eine Aktion aufgrund des Events nicht zuzulassen Anwendung z.B. bei RESET und SUBMIT vom Formular
<code>.screenX</code>	nur IE ab 4.x horizontale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms IE ab 4.x und NS ab 4.x
<code>.screenY</code>	vertikale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms IE ab 4.x und NS ab 4.x
<code>.shiftKey</code>	true, so Shift-Taste gedrückt (egal ob linke oder rechte) nur IE ab 4.x
<code>.shiftLeft</code>	true wenn linke Shift-Taste gedrückt, sonst false nur IE ab 5.5 Hinweis: rechte shift-Taste ermitteln aus <code>(shiftKey && (!shiftLeft))</code>
<code>.srcElement</code>	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per <code>onXXX = ""</code> oder Bezug auf den logischen Namen des HTML-Elementes <code>if (logischer_name == event.srcElement)</code>
<code>.srcFilter</code>	nur IE ab 4.x Zeiger auf das Filter-Objekt, für das das Ereignis ausgelöst wurde nur verwenden innerhalb <code>onfilterchange</code> -Handler nur IE ab 4.x



.target	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per onXXX = "" oder Bezug auf den logischen Namen des HTML-Elementes if (logischer_name = Ereignis.target)		
	nur NS ab 4.x		
.toElement	enthält Zeiger desjenigen Objektes, das Ereignis onmouseover hat nur IE ab 4.x siehe auch .fromElement		
.type	enthält die Event-Art z.B.	abort	
	IE ab 4.x und NS ab 4.x		
.which	enthält bei gedrückter	Maustaste:	1 für linke Taste 2 für mittlere Taste 3 für rechte Taste
		Tastatur	ASCII-Code der Taste Umwandlung in String per String.fromCharCode(TastenKodeASCII)
	nur NS ab 4.x		
.width	Breite in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde nur NS		
.x	bei IE ab 4.x: horizontale Pixel-Position der Maus bezüglich der linken oberen Ecke (0,0) des Eltern-HTML-Elementes (parent)		
	bei NS ab 4.x : identisch mit .layerX		
.y	bei IE ab 4.x: vertikale Pixel-Position der Maus bezüglich der linken oberen Ecke (0,0) des Eltern-HTML-Elementes (parent)		
	bei NS ab 4.x: identisch mit .layerY		

4.3.2.2.5.3. Methoden im IE und NS

keine

Events können zusätzlich nur durch Methoden anderer Objekte verwaltet werden.

4.3.2.2.5.4. event Objekt des Internet Explorer

Standardgemäß gilt: Kinder reichen Events zu den Eltern hoch und das bis zum BODY, also der obersten Dokumentenebene. Die Eltern haben standardgemäß immer das letzte Wort: Will ein Kind ein Event auslösen, so kann es das. Aber die Eltern entscheiden, ob das Event erhalten bleibt. Besonders gilt das für Body (document.body).

Beispiel: Wird per

```
document.body.onclick=OnClickHandler; // ohne () kodieren, damit nicht sofort ausgeführt wird
document.body.attachEvent('onclick',OnClickHandler);
```

ein Eventhandler instanziiert, so kann das Kind zwar **sein** onclick (eigener Handler muss im Kind implementiert sein) auslösen und entsprechende Reaktionen verursachen, aber die Eltern, also document.body, können innerhalb OnClickHandler() das Ereignis onclick sperren, so dass die Reaktionen des Kindes aufgehoben werden. Diese Reaktionen sind also nur solange wirksam, bis document.body das Event onclick erreicht hat. Fatal wird das bei Bildschirmausgaben des Kindes aufgrund onclick: Diese Ausgaben werden schlichtweg durch die Eltern aufgehoben (keine Anzeige mehr im Dokument).

Beispiel: Wird per

```
function OnContextMenuHandler()
// Unterbindung des Kontextmenüs
// Achtung: return false; unterdrückt nicht das Kontextmenü
{event.returnValue=false;}

document.body.oncontextmenu=OnContextMenuHandler;// ohne () kodieren, damit nicht sofort ausgeführt wird
document.body.attachEvent('oncontextmenu',OnContextMenuHandler);
```

ein Eventhandler zum Ereignis oncontextmenu implementiert, so bleibt der rechte Maustastendruck im gesamten Dokument und für alle Kinder wirkungslos, es sei denn, das Kind implementiert einen eigenen Handler für das Ereignis des Drückens der rechten Maustaste (z.B. per Event onmousedown).

4.3.2.2.5.4.1. Zugriff

event.eigenschaft
event.methode()

window.event.eigenschaft
window.event.methode()

window kann entfallen wenn aktuelles Fenster gemeint ist

zeiger_auf_fenster.event.eigenschaft
zeiger_auf_fenster.event.methode()

4.3.2.2.5.4.1. Eigenschaften

.altKey ALT-Tasten-Status
.altLeft linke ALT-Taste-Status



	nicht für Win9x
	nur für Dokument, das den Fokus hat
.button	Maustaste die das Event auslöst NUR per onmousedown, onmouseup, und onmousemove events
.cancelBubble	Event durchreichen über Eventhandlerkette
.clientX	X-Koordinate Maus relativ zum Fenster aber ohne Rahmen, Scrollbar etc.
.clientY	Y-Koordinate Maus relativ zum Fenster aber ohne Rahmen, Scrollbar etc.
.ctrlKey	CTRL-Tasten-Status
.ctrlLeft	linken CTRL-Taste Status
	nicht für Win9x
	nur für Dokument, das den Fokus hat
.fromElement	Referenz auf Maus-Eventauslösendes Quell-Element bei Mausbewegung
	nicht für Event ondragleave
.keyCode	Unicode der Taste, die das Event auslöst per onkeydown, onkeyup und onkeypress
.offsetX	X-Koordinate Maus relativ zum Objekt, das das Event auslöst
.offsetY	Y-Koordinate Maus relativ zum Objekt, das das Event auslöst
.propertyName	Name der Eigenschaft, die wertmässig verändert wurde durch onpropertychange Event auch Style-Eigenschaft etc.
.returnValue	Returnwert für den Eventhandler festlegen anstelle von return-Anweisung Achtung: Wenn kodiert, so wird eine ebenfalls im Eventhandler kodierte die Anweisung return;
	ignoriert
.screenX	X-Koordinate Maus relativ zum Bildschirm
.screenY	Y-Koordinate Maus relativ zum Bildschirm
.shiftKey	SHIFT-Tasten-Status
.shiftLeft	SHIFT-Taste links Status
	nur für Dokument mit Fokus
.srcElement	Referenz auf Objekt das das Event auslöst
.toElement	Referenz auf Maus-Eventauslösendes Ziel-Element bei Mausbewegung
	nicht für Event ondragleave
.type	Bezeichner des Events, also Eventart
.URL	Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-Datei von Element auf der Timeline es muss Ereignis onURLFlip aufgetreten sein siehe Objekt currTimeState und Behavior .style.time2
.wheelDelta	liefert Umdrehung und Richtung in der das Mousrad gedreht wurde Verwendung bei Event onmousewheel ab IE 6.x
.x	X-Koordinate Maus relativ zum Eltern-Objekt ab IE 5.x zum Fenster unter IE 5.x X-Koordinate ist relativ zum BODY-Element wenn Objekt, das das Event auslöst, absolut positioniert ist (z.B. per Style) oder Eltern nicht absolut positioniert sind
.y	Y-Koordinate Maus relativ zum Eltern-Objekt ab IE 5.x zum Fenster unter IE 5.x Y-Koordinate ist relativ zum BODY-Element wenn Objekt, das das Event auslöst, absolut positioniert ist (z.B. per Style) oder Eltern nicht absolut positioniert sind

4.3.2.2.5.4.2. Methoden

keine

Methoden anderer Objekte zur Verwaltung von Events:

.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.createEventObject()	Event-Objekt im Dokument erzeugen nur für Methode .fireEvent()
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()



4.3.2.2.5.4.3. event.bookmarks Collection

Feld der Zeiger auf ActiveX Data Objects (ADO)-Bookmarks

Syntax:

```
[ var ZeigerAufFeld = ] object.event.bookmarks
[ var ZeigerAufFeldElement = ] object.event.bookmarks[Index]
```

object kann entfallen wenn aktuelles Fenster gemeint ist, also window
zeiger_auf_fenster

Index Integer ab 0
oder String Name oder ID des Elementes
muss in [] kodiert sein

Eigenschaften:**.length**

Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:**.item()**

Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

4.3.2.2.5.4.4. event.dataTransfer Objekt des Internet Explorer

Objekt für Drag und Drop zwischen Quelle und Ziel über Clipboard (Zwischenablage)

ab IE 5.x

siehe auch Objekt clipboardData des IE

Drag = aus Quelle verschieben: Objekt mit Maus anklicken/selektieren, dann ziehen bei gedrückter
Maustaste
oder Objekt mit Maus markieren und per Tastatur CTRL-X

kopieren bzw. ausschneiden:
Objekt mit Maus selektieren, dann per Menü der rechten Maustaste kopieren
bzw. ausschneiden
oder Objekt mit Maus markieren und per Tastatur CTRL-C bzw. CTRL-X

Drop = im Ziel ablegen

nach verschieben aus Quelle:
gezogenes Objekt über Objekt ablegen durch loslassen der Maustaste
keine Mehrfachablage

nach kopieren bzw. ausschneiden aus Quelle:
auf dem Zielobjekt per Menü der rechten Maustaste in das Ziel einfügen
Mehrfacheinfügung möglich

Tastatur: über dem Ziel CTRL-V
Mehrfacheinfügung möglich

Für die Verwaltung der Zwischenablage wird das Event-Objekt mit den Ereignissen ondragXXX benutzt.

Standard-Drag und Drop (Standard-Eventhandler) bei folgenden HTML-Objekten
A, IMG, TEXTAREA, INPUT TYPE=text

Für Elemente ohne Standard-Drag und Drop müssen Eventhandler für Drag und Drop **programmiert** werden.

Das Clipboard funktioniert nur innerhalb ein und derselben Domain,
also nicht per HTTP
HTTPS
FTP etc.
nur innerhalb ein und derselben Browserinstanz
nicht zwischen Browser und externe Anwendungen z.B. MS Word

verwaltet folgende Datenformate: Text, URL, File, HTML, Image

DIV-Objekt und IFRAME-Objekt unterstützen den Datentransfer mit folgenden Attributen:

<u>HTML-Attribut</u>	<u>Eigenschaft</u>
DATAFLD	.dataFld
DATAFORMATAS	.dataFormatAs
DATASRC	.dataSrc
-----	.recordNumber

Syntax:

```
object.event.dataTransfer.eigenschaft
object.event.dataTransfer.methode()
```

object kann entfallen wenn aktuelles Fenster gemeint ist, also window



zeiger_auf_fenster

Beispiel 1:

```

<HEAD>
<SCRIPT>
    function DragStart() // Url eines Ankers als Datenformat festlegen und Daten holen
    { event.dataTransfer.setData("URL", ID_A.href);}

    function DragEnde() // Daten im URL-Format als Textdaten in den Span ablegen
    {ID_Span.innerText = event.dataTransfer.getData("URL");}
</SCRIPT>
</HEAD>
<BODY>
    <A ID="ID_A" HREF="anker"
        onclick="return(false);"
        ondragstart=" DragStart()"
    >
        Testanker
    </A>
    <SPAN ID="ID_Span" ondragenter=" DragEnde()">
        obigen Link auf diese Stelle hierher dropfen
    </SPAN>
</BODY>

```

Beispiel 2:

```

<HEAD>
<SCRIPT>
    function InitiateDrag()
    { event.dataTransfer.setData("URL", ID_Image.src);}

    function FinishDrag()
    {ID_Span.innerText = event.dataTransfer.getData("URL");}
</SCRIPT>
</HEAD>
<BODY>
    <IMAGE ID="ID_Image" SRC="/test/graphics/test.gif"
        ondragstart="InitiateDrag()">
    <SPAN ID="ID_Span" ondragenter="FinishDrag()"
    >
        Image hierher dropfen
    </SPAN>
</BODY>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    var Wert;

    function TextBereichErzeugen()
    {
        // Text im gesamten Body adressieren
        var TextBereich = document.body.createTextRange();

        // davon den Textteil des Source-Div adressieren
        TextBereich.findText(ID_DivSource.innerText);

        // und diesen selektieren
        TextBereich.select();
    }

    // Ausschneiden aktivieren, da standardgemäß für DIV deaktiv ist
    // Ausschneiden bedeutet: Browser verschiebt automatisch in das Clipboard
    function AusschneidenAktivieren() // ist Eventhandler für onbeforecut
    { event.returnValue = false; } // Event-Handler-Rückkehrcode

    function Ausschneiden() // ist Eventhandler für oncut
    {
        // Clipboard füllen mit Daten vom Texttyp
        // als Text des Source-Div
        Wert = window.clipboardData.setData("Text", ID_DivSource.innerText);
    }

```



```

// Source-Div Textbereich löschen
ID_DivSource.innerText = "";

// Rückgabewert vom Clipboardfüllen als Text (true oder false)
//      im Text des Input-Button anzeigen
ID_Input.innerText += Wert; // Wert mit als Text

// Event-Handler-Rückkehrcode
event.returnValue = false;
}
// Einfügen aktivieren, da standardgemäß für DIV deaktiviert ist
// Einfügen bedeutet: Browser fügt aus Clipboard in das Zielobjekt ein
function EinfuegenAktivieren() // ist Eventhandler für onbeforepaste
{ event.returnValue = false; } // Event-Handler-Rückkehrcode

function Einfuegen() // ist Eventhandler für onpaste
{
    // aus Clipboard Daten vom Texttyp holen und in den Textbereich
    // des Target-Div ablegen
    ID_DivTarget.innerText = window.clipboardData.getData("Text");

    // Event-Handler-Rückkehrcode
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY onload="TextBereichErzeugen()">
  <DIV ID="ID_DivSource" onbeforecut="AusschneidenAktivieren()"
    oncut="Ausschneiden()"
  >
    Diesen Text mit Maus markieren und ausschneiden
  </DIV>
  <DIV ID="ID_DivTarget" onbeforepaste="EinfuegenAktivieren()"
    onpaste="Einfuegen()"
  >
    An diese Stelle den ausgeschnittenen Text einfügen
  </DIV>
  <BR>
  <INPUT ID="ID_Input" TYPE="text" VALUE="Rückkehrcode = ">
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML>
<HEAD>
<SCRIPT>
// ##### Quellobjekt #####
function EventHandlerFuerOnDragStart()
// Quellobjekt löst Event aus:
//      in Quelle wird markiert und selektiert
{
    // selektierte Quell-Daten in die Zwischenablage puffern
    //      (Puffer wird per window.event-Objekt verwaltet)
    //      Datentyp ist hier uninteressant, da für die Zwischenablage
    //      der Typ automatisch erkannt wird, also
    //      Speicherung der Daten in der Zwischenablage
    //      immer typgerecht ist
    var ZwischenAblage = window.event.dataTransfer;

    // und diese Daten als verschiebbar erklären:
    //      aus der Quelle in die Zwischenablage
    ZwischenAblage.effectAllowed = "move";
}

// ##### Zielobjekt #####
function EventHandlerFuerOnDragEnter()
// Zielobjekt löst Event aus
// Maus über Ziel nach dem Draggen der Daten,
//      UND Maustaste ist noch nicht losgelassen
// also Zielobjekt wird mit den Daten betreten
{
    // Daten adressieren
    var ZwischenAblage = window.event.dataTransfer;

```



```

        // und diese Daten als verschiebbar erklären:
        //      aus der Zwischenablage in das Zielobjekt
        ZwischenAblage.dropEffect = "move";

        // Event-Handler-Rückkehrcode
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }

    function EventHandlerFuerOnDrop
    // Zielobjekt löst Event aus
    // Maus über Ziel nach dem Droppen der Daten,
    //      UND Maustaste wird losgelassen
    {
        // Ziel adressieren, das mit ondrop-Ereignis behandelt werden soll
        var Ziel = window.event.srcElement;

        // Daten in der Zwischenablage adressieren und holen
        //      (Puffer wird per window.event-Objekt verwaltet)
        var ZwischenAblage = window.event.dataTransfer;

        // und Daten vom Texttyp aus der Zwischenablage im Ziel ablegen,
        //      da Ziel nur Textdaten empfangen kann
        Ziel.innerText += Daten.getData("text");

        // Event-Handler-Rückkehrcode
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }

    function EventHandlerFuerOnDragOver
    // Zielobjekt löst Event aus
    {
        // nichts tun ausser Rückkehrcode des Handlers liefern
        var EventObjekt = window.event;
        EventObjekt.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ondragstart=" EventHandlerFuerOnDragStart ()"
    >
        Diesen Text markieren und draggen
    </DIV>
    <BR>
    <DIV
        ondragover="EventHandlerFuerOnDragOver()"
        ondragenter="EventHandlerFuerOnDragEnter()"
        ondrop="EventHandlerFuerOnDrop()"
    >
        An diese Stelle den Text dropfen
    </DIV>
</BODY>
</HTML>

```

Beispiel 5 für Clipboard-Nutzung ohne Drag & Drop:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var TextBereich = document.body.createTextRange();
        TextBereich.findText(ID_Div1.innerText);
        TextBereich.select(); // selektieren
    }

    function VorDemAusschneiden()
    {event.returnValue = false;} // Cut per Menü aktivieren

    function Ausschneiden()
    {
        ID_Div1.innerText = "";
        ID_Input.innerText += window.clipboardData.setData("Text", ID_Div1.innerText);
    }

```



```

// true oder false
// Clipboard-Daten sind Texttyp
event.returnValue = false;
}

function VorDemEinfuegen()
{ event.returnValue = false; } // Paste per Menü aktivieren

function Einfuegen()
{
    ID_Div2.innerHTML = window.clipboardData.getData("Text");
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" onbeforecut="VorDemAusschneiden()" oncut="Ausschneiden()">
        Dieses Text selektieren und ausschneiden
    </DIV>
    <DIV ID="ID_Div2" onbeforepaste="VorDemEinfuegen()" onpaste="Einfuegen()">
        den ausgeschnittenen Text hier einfügen
    </DIV><BR>
    <INPUT ID="ID_Input" TYPE="text" READONLY VALUE="" SIZE="6">
</BODY>
</HTML>

```

Eigenschaften:

.dropEffect Cursor-Layout bei Drop
 wird von Eigenschaft .effectAllowed beeinflusst
 .effectAllowed Art von Drag und Drop
 beeinflusst Eigenschaft .dropEffect

Methoden:

.clearData() Clipboardinhalt löschen
 .getData() Clipboard auslesen
 .setData() Clipboard füllen, also Daten dort ablegen
 wenn Clipboard nicht leer so immer anhängen

4.3.2.2.5.4.5. Eventarten (Auswahl)

Hinweis zum Test von Mausereignissen: Bitte bei forlaufenden Ereignissen wie onmousemove etc. **kein** alert() verwenden, da das Betreten/Betätigen der Alert-Box selbst das Ereignis erzeugen kann. Anstelle dafür einen DIV verwenden, dessen innerHTML bzw. innerText per gleichnamige Eigenschaften belegt wird zum Mausereignis und so keine zusätzlichen Mausereignisse auftreten können. Alternativ ist auch die Fenster-Status-Zeile belegbar.

onabort Abbruch des Download vom Image
 onactivate erzeugt wenn Objekt als aktives gesetzt wird (event.fromElement to the event.srcElement)
 aktiv setzen bedingt nicht den Erhalt des Focus:
 aber Aktivierung per Fokus-Methode möglich
 wird immer direkt vor onload erzeugt
 ab IE 5.5

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    function EventOnActivate()
    {ID_Div.innerHTML += "onactivate erkannt";}

    function EventOnLoad()
    {ID_Div.innerHTML += "onload erkannt";}
</SCRIPT>
</HEAD>
<BODY onactivate="EventOnActivate();" onload="EventOnLoad();">
    <DIV ID="ID_Div"></DIV>
</BODY>
</HTML>

```

onafterprint erzeugt mit Ende des Druck bzw. Druckvorschau
 onafterupdate erzeugt wenn Daten in einem Datasource-Objekt erfolgreich geupdatet wurden
 onbeforeactivate erzeugt direkt vor der Aktivierung eines Objektes
 onbeforecopy erzeugt direkt vor dem Kopieren einer Selektion im Objekt in das Clipboard

Beispiel

```

<HEAD>
<SCRIPT>
    var Daten= "Das sind Text-Daten";

    function Quelle_Beforecopy()
    {event.returnValue = false; }

```




```

function Quelle_Copy()
{window.clipboardData.setData("Text", Daten); }

function Ziel_BeforePaste()
{event.returnValue = false; }

function Ziel_Paste()
{
    ID_Input.value = window.clipboardData.getData("Text");
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecopy="Quelle_Beforecopy()"
        oncopy="Quelle_Copy()"
    >
        diesen Text kopieren
    </SPAN>
    <INPUT ID="ID_Input" onbeforepaste="Ziel_BeforePaste()"
        onpaste="Ziel_Paste()"
    >
</BODY>

```

onbeforecut erzeugt direkt vor dem Ausschneiden einer Selektion im Objekt

Beispiel 1

```

<HEAD>
<SCRIPT>
    var Kette = "";

    function EventBeforeCut()
    {event.returnValue = false; }

    function EventCut()
    {
        Kette= ID_Span.innerText;
        ID_Span.innerText = "";
        event.returnValue = false;
    }

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Div.innerText = Kette;
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        diesen Text selektieren und ausschneiden
    </SPAN>
    <BR>
    <DIV ID="ID_Div" onbeforepaste="EventBeforePaste()" onpaste="EventPaste()">
        hierher den ausgeschnittenen Text einfügen
    </DIV>
</BODY>

```

Beispiel 2

```

<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";

    function Init()
    {
        var TextBereich = document.body.createTextRange(); // Bereich aller Texte
        TextBereich.findText(ID_Span1.innerText);           // aber den Text im ID_Span1 suchen
        TextBereich.select();                                // und dann selektieren
    }

    function EventBeforeCut()
    {
        Kette = ID_Span1.innerText;
    }

```



```

        event.returnValue = false;
    }

    function EventCut()
    {window.clipboardData.setData("Text", TextDaten); }           // setData liefert return-Wert
                                                                    //      also
    event.returnValue = ...;                                       //      noch nötig zu
                                                                    //      kodieren

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID=" ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfuegen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

onbeforedeactivate erzeugt direkt bevor ein Objekt als deaktiv gesetzt wird
 onbeforeeditfocus erzeugt direkt vor der User-Interaktivität auf ein editierbares Objekt
 immer erzeugt wenn INPUT-Objekt oder TEXTAREA-Objekt den focus erhält
 onbeforepaste erzeugt direkt vor dem Einfügen aus dem Clipboard in ein Objekt
 onbeforeprint erzeugt direkt vor dem Druck bzw. Druckvorschau
 onbeforeunload erzeugt direkt vor dem Unload eines Dokumentes
 unload = verlassen des Dokumentes durch Laden eines neuen Dokumentes in das selbe Fenster

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    function DokumentSchliessen()
    {event.returnValue = "Irgendeinen Stringwert liefern. Vor dem Schliessen wird Dialogbox angezeigt."}
</SCRIPT>
</HEAD>
<BODY onbeforeunload="DokumentSchliessen ()">
    <A HREF="http://www.test.de">zu www.test.de</A>
</BODY>
</HTML>

```

onbeforeupdate erzeugt mit dem Beginn des Update von Daten eines Datasource-Objektes
 onbegin erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird
 nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element
 erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element
 siehe onend und onrepeat
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL ID="ID_Excl"
        BEGIN="indefinite"
        DUR="5"
        REPEATCOUNT="5"
        onbegin="alert('Start der Animation');">
    <t:ANIMATEMOTION ID="ID_Animatemotion"
        TARGETELEMENT="ID_Div"

```



```

        TO="200,0"
        BEGIN="0"
        DUR="2"
        AUTOREVERSE="true"
    >
    </t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
        border:solid black 1px;
    "
>
    sich bewegender DIV
</DIV>
<BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
<BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
</BODY>
</HTML>

```

onblur erzeugt wenn Objekt den Focus verliert

Beispiel

```

<HTML>
<BODY>
    <INPUT TYPE=text NAME="Test" VALUE="onblur Test" onblur="alert(event.srcElement.name)">
</BODY>
</HTML>

```

onbounce erzeugt wenn Behavior-Eigenschaft des Marquee-Objektes auf "alternate" gesetzt ist
und der Marquee-Text eine der beiden Seiten des Elternfensters erreicht hat

Beispiel

```

<BODY>
    <MARQUEE BEHAVIOR="alternate" WIDTH=200 LOOP=3
        onbounce="alert('onbounce-Ereignis erkannt')">
    >
        Marquee Text
    </MARQUEE>
</BODY>

```

oncellchange erzeugt wenn Daten verändert wurden in der Datenquelle z.B. Objekt
onclick erzeugt mit Druck der linken Maustaste auf ein Objekt
benötigt vorher die Eventfolge erzeugt onmousedown und onmouseup
wobei die Maus auch dabei über dem Objekt sein muss
bei Radio-Buttons-Gruppe: onclick event immer nach
onbeforeupdate und onafterupdate events for the control group.
bei focusfähigem Objekt: onfocus event erzeugt vor dem onclick event
bei Doppelklick per linker Maustaste:
zuerst onclick ausgelöst, dann ondblclick

oncontextmenu erzeugt wenn rechte Maustaste gedrückt wurde
Unterdrückung, wenn der Eventhandler event.returnValue=false; enthält
(mit return false; erfolgt keine Unterdrückung)

Beispiel

```

<SPAN STYLE="width:300; background-color:blue; color:white;" oncontextmenu="return false">
    <P>Das Kontextmenue ist innerhalb dieses SPAN abgeschaltet</P>
</SPAN>

```

oncontrolselect erzeugt wenn User eine Control-Selektion im Objekt tätigt (z.B. CTRL+ Maus)
oncopy erzeugt wenn Quellelement oder Selektion dem Clipboard hinzugefügt wird
per rechte Maus-Click und Anwahl des Menüpunktes Copy
Hinweis: Festlegung der Datenart per Methode .setData()
oder CTRL-C

oncut erzeugt durch Quell-Objekt wenn Daten aus Quellobjekt in das Clipboard verschoben werden

Beispiel 1

```

<HEAD>
<SCRIPT>
    var Kette = "";

    function EventBeforeCut()
    {event.returnValue = false; }

    function EventCut()
    {
        Kette= ID_Span.innerText;
        ID_Span.innerText = "";
        event.returnValue = false;
    }

```



```

function EventBeforePaste()
{event.returnValue = false; }

function EventPaste()
{
    ID_Div.innerText = Kette;
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Span" onbeforecut="EventBeforeCut()"oncut="EventCut()">
    diesen Text selektieren and ausschneiden
</SPAN>
<BR>
<DIV ID="ID_Div" onbeforepaste="EventBeforePaste()"onpaste="EventPaste()">
    hierher den ausgeschnittenen Text einfügen
</DIV>
</BODY>

```

Beispiel 2

```

<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";

    function Init()
    {
        var TextBereich = document.body.createTextRange(); // Bereich aller Texte
        TextBereich.findText(ID_Span1.innerText);           // aber den Text im ID_Span1 suchen
        TextBereich.select();                               // und dann selektieren
    }

    function EventBeforeCut()
    {
        Kette = ID_Span1.innerText;
        event.returnValue = false;
    }

    function EventCut()
    {window.clipboardData.setData("Text", TextDaten); }      // setData liefert return-Wert
                                                           // also
    event.returnValue = ...;                                // noch nötig zu
                                                           //
    kodieren

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()"oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID=" ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfuegen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

ondataavailable	erzeugt von der Datenquelle bei jedem Senden von Daten (asynchrones Senden von der Datenquelle (Objekt))
ondatasetchanged	erzeugt von der Datenquelle wenn Daten geändert wurden in der Datenquelle verwendet bei Filter-Operationen
ondatasetcomplete	erzeugt von Datenquelle wenn alle Daten der Quelle verfügbar also sendbar sind
ondblclick	erzeugt wenn Doppelklick mit Maus mit folgender Eventfolge onmousedown, onmouseup, onclick, onmouseup, und then ondblclick.
ondeactivate	erzeugt mit Deaktivierung eines Objektes
ondrag	erzeugt kontinuierlich während allen Drag-Operationen, aber erst nach ondragstart Event



ondragend	<p>Drag = Maus gedrückt halten</p> <p>erzeugt am Ende der Dragoperation also wenn Maus losgelassen wird</p> <p>erzeugt immer vom Quellobjekt</p> <p>Hinweis: Zielobjekt erzeugt anschließend ondragleave Event</p>
ondragenter	<p>erzeugt wenn Maus über Objekt gedrückt wurde UND der User das Quell-Objekt bei gedrückter Maustaste zieht</p> <p>erzeugt immer vom Ziel-Objekt wenn Ziel erreicht wurde</p> <p>ist das ERSTE Event des Ziels, also das als Erstes ausgelöste Event</p> <p>erzeugt immer vom Quell-Objekt solange Ziel nicht erreicht wurde</p> <p>wird immer direkt vor dem ondragover Event erzeugt</p>
ondragleave	erzeugt vom Ziel-Objekt wenn User die Maus aus dem Ziel bewegt während Drag-Operation
ondragover	erzeugt vom Ziel-Objekt wenn User die Maus über das Ziel bewegt während Drag-Operation UND immer direkt nach dem ondragenter Event
ondragstart	<p>erzeugt vom Quell-Objekt wenn User die Selektion für Drag-Operationen ausführt</p> <p>auch bei Textelement</p> <p>ist ERSTES Event für Drag-und Drop</p> <p>Hinweis: Quell-Objekt nutzt Methode .setData() für Übergabe der Daten</p>
ondrop	<p>erzeugt vom Ziel-Objekt wenn Maustaste losgelassen wird über dem Ziel</p> <p>Drop = Ablegen</p> <p>direkt erzeugt vor ondragleave und ondragend Events.</p>
onend	<p>erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive aller Wiederholungen laut .repeatCount</p> <p>bzw. wenn das aktive Element gestoppt wird</p> <p>erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat</p> <p>und somit des Kindelement ebenfalls endet</p> <p>nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde, es sei denn, das Elternelement hat das Ende seiner Timeline erreicht</p> <p>siehe Methode .endElement() und Eigenschaft .end</p> <p>siehe onbegin und onrepeat</p> <p>siehe Objekt currTimeState und Behavior .style.time2</p>

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="indefinite"
DUR="5"
REPEATCOUNT="5"
onend="alert('Ende der Animation');">
<t:ANIMATEMOTION ID="ID_Animatemotion"
TARGETELEMENT="ID_Div"
TO="200,0"
BEGIN="0"
DUR="2"
AUTOREVERSE="true">
</t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
border:solid black 1px;
">
</DIV>
<BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
<BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
</BODY>
</HTML>
```

onerror erzeugt wenn Laufzeit-Fehler beim Laden eines Objektes

Beispiel 1:

```
<SCRIPT ...>
<!--
function eigenes_fehler_fenster(meldungs_text,url,zeilen_nr)
{
// Fehlermeldung bilden
var url_als_kette=url.toString();
```



```

var zeilen_nr_als_kette=zeilen_nr.toString();
var meldung_komplett=medlungs_text + url_als_kette + zeilen_nr_als_kette;

// Meldungsfenster
var fenster=window.open();
with (fenster.document)
{
    open("text/html"); // HTML-Dokument im Fenster erzeugen

    writeln("<HTML><HEAD><TITLE>Private Errormeldung</TITLE></HEAD>");
    writeln("<BODY><H1>Fehlermeldung</H1>");

    writeln(meldung_komplett);

    writeln("<FORM><INPUT TYPE=BUTTON VALUE='OK'
onClick='self.close()'>");
    ); // ist EINE Zeile

    // Button anklicken, damit das Meldungs-Fenster geschlossen wird
    close(); //HTML-Dokument schliessen
}
return true; // muss true liefern für window.onerror;
}

var RetteOnError=window.onerror;

window.onerror= eigenes_fehler_fenster; // NICHT onError kodieren !
// keine () kodieren, da sonst die Funktion
// sofort ausgeführt wird !!
// window.onerror verlangt die Zuweisung
// von true für Abschaltung des
// Standard-onerror

// -->
</SCRIPT>
...
<BODY>
...
</BODY>

```

Beispiel 2:

```

<SCRIPT>
function FehlerAufspueren (MeldungString, UrlString, ZielenNummerString)
{
    ID_Div.innerHTML ="<B>Fehler erkannt</B>";
    ID_Div.innerHTML+="Error: " + MeldungString      + "<BR>";
    ID_Div.innerHTML+="Line: " + ZielenNummerString + "<BR>";
    ID_Div.innerHTML+="URL: " + UrlString             + "<BR>";

    return false;
}

window.onerror=FehlerAufspueren; // ohne () kodieren
</SCRIPT>
<BODY>
<DIV ID="ID_Div"></DIV>
<BODY>

```

Beispiel 3 für Bild:

```

<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette; // wird sofort geparkt, also IMG-Tag ausgeführt
                           // ( anstelle von eval()
                           // bzw. document.write() )

    ID_IMG.src="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

```



	<pre> function IMGAltTextAufFehlerMeldung () { ID_IMG.alt="Das Bild konnte nicht geladen werden."; return true; } </SCRIPT> <INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()"> <DIV ID="ID_Div"></DIV> </pre>
onerrorupdate	erzeugt wenn Fehler während Datenupdate für ein Datasource-Objekt auftrat
onfilterchange	erzeugt wenn ein visueller Filter komplett abgelaufen ist
onfinish	erzeugt wenn alle Durchläufe des Marquee-Objektes komplett beendet sind
	Anzahl der Durchläufe laut LOOP-Attribut
	LOOP-Attribut muss Wert > 1 haben, wenn onfinish erzeugt werden soll
Beispiel	<pre> <BODY> <MARQUEE LOOP=2 onfinish="alert(event.srcElement.id + ' onfinish-Ereignis erkannt')"> Marquee Text </MARQUEE> </BODY> </pre>
onfocus	erzeugt, wenn Objekt den Focus erhält
	Fokus nur erhaltbar nach dem kompletten Laden des Dokumentes
Beispiel für Label	<pre> <STYLE> .normal {background-color:"#FFFFFF"; color:"#000000"; font-weight:normal; font-size:8pt; font-family:Arial;} .accessible { background-color:"beige"; font-weight:bold; font-size:10pt;} </STYLE> <SCRIPT> function StyleWechsel() { // Input-Objekt event.srcElement.className="accessible"; // Input-Objekt // Label-Objekt var ZeigerAuf Label =eval(event.srcElement.id + "_Label "); // ID ist "ID_Input" // also "ID_Input" + "_Label" = "ID_Input_Label" // Zeichenkettenoperation leider nötig, wenn mehrere // Objekte mit Eventerzeugung vorhanden wären // und ebenfalls nötig wegen Bezug im Label auf das // ID des Objektes, das Label haben soll ZeigerAuf Label.className="accessible"; } </SCRIPT> <LABEL FOR="ID_Input" oInput" CLASS="normal" ID="ID_Input_Label">Text eingeben</LABEL> <INPUT TYPE="text" CLASS="normal" onfocus="StyleWechsel()" ID="ID_Input"> </pre>
onfocusin	erzeugt bevor das Element den Focus erhält
onfocusout	erzeugt nachdem das Element den Focus verloren hat
onhelp	erzeugt wenn F1-Taste gedrückt im aktuellen Fenster
onhide	erzeugt wenn der Media Bar Player gerade unsichtbar gemacht wird (versteckt wird)
	siehe Eigenschaft .enabled
	entspricht dem Schliessen des Media Bar Player durch den User
	Media Bar Player ist der Windows Media Player
	siehe Behavior .style.mediaBar
onkeydown	erzeugt wenn irgend eine Tastatur-Taste gedrückt wurde
Beispiel	<pre> <SCRIPT> function TastenCodeHolen() { if(ID_Input.checked) { ID_Textarea.innerText+="[Keycode = " + event.keyCode + "]\n"; event.returnValue=false; } else { ID_Textarea.innerText+=String.fromCharCode(event.keyCode); } } </SCRIPT> <INPUT TYPE="checkbox" ID="ID_Input"> <INPUT TYPE="text" onkeydown=" TastenCodeHolen()"> <TEXTAREA ID="ID_Textarea" ROWS="10" COLS="50"></TEXTAREA> </pre>
onkeypress	erzeugt bei Druck einer alphanumerischen Tastatur-Taste
Beispiel	<pre> <HEAD> </pre>



```

<SCRIPT>
    function ShiftPruefen()
    {
        if (window.event.shiftKey)
        {ID_Input.value = "Shift erkannt";}
    }
</SCRIPT>
</HEAD>
<BODY>
    drücke SHIFT mit anderer Taste
    <INPUT TYPE=text onkeypress="ShiftPruefen()">
    <INPUT TYPE=text ID="ID_Input">
</BODY>
onkeyup          erzeugt wenn gedrückte Tastatur-Taste losgelassen wird
onlayoutcomplete erzeugt wenn das Druckobjekt bzw. Druckvorschau-Objekt komplett gefüllt ist
onload           setzt Eigenschaft .contentOverflow auf true
                 erzeugt mit dem Laden eines Objektes

    Beispiel 1
    <BODY>
        <SCRIPT FOR=window EVENT=onload LANGUAGE="JScript">
            window.status = "Seite ist geladen!";
        </SCRIPT>
    </BODY>

    Beispiel 2
    <SCRIPT>
        function Meldung()
        {window.status = "Image " + event.srcElement.src + " ist geladen";}
    </SCRIPT>
    <BODY>
        <IMG SRC="test.gif" onload="Meldung()">
    </BODY>
onlosecapture    erzeugt, wenn Objekt die Mausüberwachung verliert

    Beispiel
    <BODY onload="ID_Div.setCapture()"
        onclick="ID_Div.releaseCapture();"
    >
        <DIV ID="ID_Div"divOwnCapture
            onmousemove="ID_Textarea.value=event.clientX + event.clientY"; // kein alert() !!!!!
            onlosecapture="alert(event.srcElement.id + ' ohne Maus-Ueberwachung');"
        >
            Ueber diesen Text mit der Maus fahren
            <BR>
            <TEXTAREA ID="ID_Textarea" COLS=2></TEXTAREA>
        </DIV>
        <DIV>
            Klick hier fuer Event onlosecapture per Methode .releaseCapture()
        </DIV>
    </BODY>
onmediacomplete erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde
                 für Animation per Timeline
                 Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer
                 Datei (streaming media file) z.B. Gif-Bild, Video etc.
                 sinnvoll für Start des Elementes auf der Timeline
                 siehe onmediaerror
                 siehe Objekt currTimeState und Behavior .style.time2

    Beispiel:
    <HTML XMLNS:t="urn:schemas-microsoft-com:time">
    <HEAD>
    <?IMPORT namespace="t" implementation="#default#time2">
    <STYLE>
        .time_line_klasse { behavior: url(#default#time2) }
    </STYLE>
    </HEAD>
    <BODY>
        <t:VIDEO          ID="ID_Video"
                        SRC="test.wmv"
                        onmediacomplete="ID_Span.innerHTML +=
                                                'Datei test.wmv wurde komplett geladen !'
                                                "
        >
        </t:VIDEO>
        <BR>
        <SPAN ID="ID_Span"></SPAN>

```




```
</BODY>
</HTML>
```

onmediaerror erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde oder das Medium einen Fehler enthält für Animation per Timeline
Media-Element benötigt vor seiner Animation den Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.
sinnvoll für Start des Elementes auf der Timeline
ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet werden darf !
siehe onmediacomplete
siehe Objekt curTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:IMG ID="ID_Img"
SRC="test.gif"
STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
onmediaerror="alert(' Datei test.gif nicht ladbar !')">
>
</t:IMG>
<t:ANIMATION ID="ID_Animation"
TARGETELEMENT="ID_Img"
TO="0,400"
DUR="2"
BEGIN="0"
ACCELERATE="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
>
</t:ANIMATION>
</BODY>
</HTML>
```

onmousedown erzeugt wenn irgendeine Maustaste gedrückt wird
onmouseenter erzeugt wenn Maus ein Objektbereich betritt
onmouseleave erzeugt wenn Maus gerade den Objektbereich verlässt
onmousemove erzeugt wenn Maus im Bereich eines Objekts bewegt wird

Beispiel

```
<SCRIPT>
function Anzeige()
{ID_Span.innerHTML="Coords: (" + event.clientX + ", " + event.clientY + ")";} // kein alert() !!!!
</SCRIPT>
<DIV onmousemove="Anzeige()">
<SPAN ID="ID_Span"></SPAN>
</DIV>
```

onmouseout erzeugt wenn Maus den Bereich eines Objektes gerade verlässt
genau 1x erzeugt pro Verlassen
onmouseover erzeugt wenn Maus den Bereich eines Objektes gerade betritt
genau 1x erzeugt pro Betreten
onmouseup erzeugt wenn Maustaste losgelassen wird
Hinweis: Eigenschaft .button liefert die losgelassene Taste
onmousewheel erzeugt wenn Mausexplorer gedreht wird
mit dem Drehen wird Eigenschaft .wheelDelta gefüllt
mit Integer-Vielfachen von 120 Grad Umdrehung
wenn > 0 so Drehung vom User weg
wenn < 0 so Drehung zum User hin

ab IE 6.0

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
var ZoomFaktorStartWert = 10; // > 0, ganzzahlig, in Prozent
var ZoomSchrittWeite = 1; // > 0, ganzzahlig
var DrehSchrittWeite = 1; // 1 oder 2 oder 3 da Faktor von 120 Grad

var ZoomFaktor = ZoomFaktorStartWert; // > 0, ganzzahlig, in Prozent
function VergrössernVerkleinern()
```



```

    {
        if (event.wheelDelta >= (DrehSchrittWeite * 120))
        { ZoomFaktor += ZoomSchrittWeite; }
        else
        {
            if (event.wheelDelta <= (-1 * (DrehSchrittWeite * 120)) )
            { ZoomFaktor -= ZoomSchrittWeite; }
        }

        if (ZoomFaktor <= 0)
        { ZooFaktor = ZoomFaktorStartWert; }

        ID_Image.style.zoom = ZoomFaktor + '0%';
    }
</SCRIPT>
</HEAD>
<BODY>
    <IMG ID="ID_Image" SRC="test.jpg" onmousewheel="VergroessernVerkleinern()">
</BODY>
</HTML>

```

onmove

erzeugt, wenn Objekt bewegt wird:
 Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute**;")
 nicht erzeugt, wenn ein Unterobjekt bewegt wird:
 Unterobjekt muss **eigenen** Handler haben

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    function HandleFuerOnMoveStart()
    {
        var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                            // falls mehrere bewegbare
                                                            // Objekte vorhanden sind

        ZeigerAufObjektMitEvent.style.backgroundColor = "green";
        ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
    }

    function HandlerFuerOnMove ()
    {
        ID_Span1.innerHTML = event.srcElement.offsetLeft;
        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandleFuerOnMoveEnd()
    {
        var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                            // falls mehrere bewegbare
                                                            // Objekte vorhanden sind

        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    document.execCommand("2D-position",false,true);    // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY>
    onmovestart="HandleFuerOnMoveStart();"
    onmove="HandlerFuerOnMove();"
    onmoveend="HandleFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offsetTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

onmoveend

erzeugt, wenn das Bewegen eines Objektes endet:
 Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute**;")
 nicht erzeugt, wenn ein Unterobjektbewegung endet:



Unterobjekt muss **eigenen** Handler haben

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
function HandleFuerOnMoveStart()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";

}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;

}

function HandleFuerOnMoveEnd()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";

}

document.execCommand("2D-position",false,true);    // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY onmovestart="HandleFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandleFuerOnMoveEnd();"
>

offsetLeft = <SPAN ID="ID_Span1"></SPAN>
<BR>
offsetTop = <SPAN ID="ID_Span2"></SPAN>
<BR>
<DIV CONTENTEDITABLE="true">
    <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
        bewegbarer DIV
    </DIV>
</DIV>
</BODY>
</HTML>
```

onmovestart erzeugt, wenn das Bewegen eines Objektes beginnt:
 Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute**;")
 nicht erzeugt, wenn ein Unterobjektbewegung beginnt:
 Unterobjekt muss **eigenen** Handler haben

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
function HandleFuerOnMoveStart()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";

}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;

}

function HandleFuerOnMoveEnd()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
```



```

//           Objekte vorhanden sind
ZeigerAufObjektMitEvent.style.backgroundColor = "red";
ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

document.execCommand("2D-position",false,true); // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY onmovestart="HandleFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandleFuerOnMoveEnd();"
>
offsetLeft = <SPAN ID="ID_Span1"></SPAN>
<BR>
offsetTop = <SPAN ID="ID_Span2"></SPAN>
<BR>
<DIV CONTENTEDITABLE="true">
    <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
        bewegbarer DIV
    </DIV>
</DIV>
</BODY>
</HTML>

```

onopenstatechange erzeugt, wenn Media Bar Player den Status bezüglich Playliste, Codec, Lizenz und Individualisierung ändert
 siehe Eigenschaft .openState
 Media Bar Player ist der Windows Media Player
 siehe Behavior .style.mediaBar

Beispiel:

```

<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
onopenstatechange="alert(ID_Div.openState);"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
"
>

```

onoutofsync erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)
 sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked":
 Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()
 siehe onsyncstored
 siehe Objekt currTimeState und Behavior .style.time2
 Syntax:

```

HTML <ELEMENT onoutofsync = "handler" ... >
Script object.onoutofsync = handler
<SCRIPT FOR = object EVENT = onoutofsync>

```

onpaste erzeugt vom Zielobjekt wenn Daten aus Clipboard in das Objekt eingefügt werden

Beispiel 1

```

<HEAD>
<SCRIPT>
var Kette = "";

function EventBeforeCut()
{event.returnValue = false; }

function EventCut()
{
    Kette= ID_Span.innerText;
    ID_Span.innerText = "";
    event.returnValue = false;
}

function EventBeforePaste()
{event.returnValue = false; }

function EventPaste()
{

```



```

        ID_Div.innerText = Kette;
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        diesen Text selektieren und ausschneiden
    </SPAN>
    <BR>
    <DIV ID="ID_Div" onbeforepaste="EventBeforePaste()" onpaste="EventPaste()">
        hierher den ausgeschnittenen Text einfügen
    </DIV>
</BODY>

```

Beispiel 2

```

<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";

    function Init()
    {
        var TextBereich = document.body.createTextRange(); // Bereich aller Texte
        TextBereich.findText(ID_Span1.innerText);           // aber den Text im ID_Span1 suchen
        TextBereich.select();                                // und dann selektieren
    }

    function EventBeforeCut()
    {
        Kette = ID_Span1.innerText;
        event.returnValue = false;
    }

    function EventCut()
    {window.clipboardData.setData("Text", TextDaten); }      // setData liefert return-Wert
                                                                // also
    event.returnValue = ...;                                  // noch nötig zu
                                                                //
    kodieren

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID="ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfügen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

onpause erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren
auch bei body Objekt
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.wmv"

```



```

onpause="ID_Span.innerText = "Pause !"
onmediacomplete="ID_Span.innerText =
    "Datei test.wmv wurde komplett geladen !"

```

```

>
</t:VIDEO>
<BR>
<SPAN ID="ID_Span"></SPAN>
</BODY>
</HTML>

```

onplaystatechange erzeugt, wenn Media Bar Player den Status bezüglich Wiedergabe ändert
 siehe Eigenschaft `.playState`
 Media Bar Player ist der Windows Media Player
 siehe Behavior `.style.mediaBar`

Beispiel:

```

<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
onplaystatechange="alert(ID_Div.playState);"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
"
>

```

onpropertychange erzeugt wenn eine Objekteigenschaft geändert wird
 außer Änderung von `.innerText` und `.innerHTML`

Beispiel

```

<HEAD>
<SCRIPT>
function ValueAendern()
{ ID_Input1.value = "Neuer Wert von VALUE";}

function FarbeAendern()
{ ID_Input2.style.backgroundColor = "aqua";}

function Anzeige()
{alert(event.propertyName + " wurde verändert");}

</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE=button ID="ID_Input1"
VALUE="Click um VALUE zu ändern"
onclick="ValueAendern()"
onpropertychange="Anzeige()"
>
<INPUT TYPE=button ID="ID_Input2"
VALUE="Click um Farbe zu ändern"
onclick="FarbeAendern()"
onpropertychange="Anzeige()"
>
</BODY>

```

onreadystatechange erzeugt wenn Status eines Objektes sich ändert
 immer erzeugt von datenladenden folgender Objekte:
 applet, document, frame, frameSet, iframe, img, link, object, script und xml elements.

Beispiel

```

function Preufen ()
{
    if (document.readyState=="complete")
    {alert('Dokument komplett geladen und mit den Daten initialisiert.);}
}

document.onreadystatechange=Preufen; // ohne () kodieren

```

onrepeat erzeugt mit Start **jeder** Wiederholung der Animation des aktiven Elementes
 nicht erzeugt beim Start des aktiven Elementes (siehe `onbegin`), also des
 ersten Durchlaufes, der keine Wiederholung ist
 nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen
 Handler für `onrepeat` kodiert hat (kein Heraufreichen des
 Ereignisses `onrepeat` zum Elternelement)



.repeatCount bzw. .repeat muss > 1 sein:
 onrepeat wird also .repeatCount - 1 mal erzeugt
 Kind eines Elementes
 siehe onend, onbegin, .repeatCount und .repeat
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:IMG ID="ID_Img"
SRC="test.gif"
STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
onmediaerror="alert('Datei test.gif nicht ladbar !')"
onrepeat="alert('Aktuelle Wiederholung: ' + ID_Animation.currTimeState.repeatCount);"
>
</t:IMG>
<t:ANIMATION ID="ID_Animation"
TARGETELEMENT="ID_Img"
TO="0,400"
DUR="2"
BEGIN="0"
ACCELERATE="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
>
</t:ANIMATION>
</BODY>
</HTML>
```

onreset

erzeugt wenn Formular zurückgesetzt wird
 Resetaktion erzeugbar per
 INPUT TYPE="reset"
 BUTTON TYPE="reset"

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
function Anzeige()
{ ID_Span.innerText += "Anzeige zum Formular Reset";}
</SCRIPT>
</HEAD>
<BODY>
<FORM NAME="Formular" ID="ID_Formular" onreset="Anzeige();"
<INPUT TYPE="text" NAME="InputText" VALUE="">
<BR>
<INPUT TYPE="reset" VALUE="Formular Reset">
<BUTTON onclick="ID_Formular.reset();">Formular Reset</BUTTON>
</FORM>
<SPAN ID="ID_Span"></SPAN>
<BR>
<BUTTON onclick="location.reload(true);">Refresh der Seite</BUTTON>
</BODY>
</HTML>
```

onreset

erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(),
 also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht
 und damit zurückgesetzt wurde
 also nicht beim Initialisieren des Elementes und seiner Timeline
 durch Instanziierung des Elementes
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="indefinite"
```



```

        DUR="5"
        REPEATCOUNT="5"
        onreset="alert('Rücksetzen der Animation und Timeline');"
    >
        <t:ANIMATEMOTION          ID="ID_Animatemotion"
                                TARGETELEMENT="ID_Div"
                                TO="200,0"
                                BEGIN="0"
                                DUR="2"
                                AUTOREVERSE="true"
        >
        </t:ANIMATEMOTION>
    </t:EXCL>
    <DIV      ID="ID_Div"
            CLASS="time_line_klasse"
            STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
                border:solid black 1px;
            "
    >
        sich bewogender DIV
    </DIV>
    <BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
    <BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
    <BUTTON onclick=" ID_Animatemotion.resetElement();">Reset</BUTTON>

</BODY>
</HTML>

```

onresize erzeugt, wenn Objektgröße verändert wird
 nicht für embedded-Objekt
 onresizeend erzeugt, wenn Änderung der Objektgröße endet
 onresizestart erzeugt wenn Veränderung Objektgröße endet
 onresume erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird
 siehe Objekt currTimeState und Behavior .style.time2
 auch für body Objekt

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                    SRC="test.wmv"
                    onpause="ID_Span.innerText = "Pause !"
                    onresume=" ID_Span.innerText = "Pause beendet ""
                    onmediacomplete="ID_Span.innerText =
                        "Datei test.wmv wurde komplett geladen !"
    >
    </t:VIDEO>
    <BR>
    <SPAN ID="ID_Span"></SPAN>
</BODY>
</HTML>

```

onreverse erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird
 (auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)
 .autoReverse muss auf "true" stehen
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL AUTOREVERSE="true"
        DUR="6"
        REPEATCOUNT="indefinite"
        onreverse="alert('Rückwärts auf der Timeline');"
    >

```




```

        <DIV CLASS="time_line_klasse" BEGIN="0" DUR="2">Zeile 1</DIV>
        <DIV CLASS="time_line_klasse" BEGIN="2" DUR="2">Zeile 2</DIV>
        <DIV CLASS="time_line_klasse" BEGIN="4" DUR="2">Zeile 3</DIV>
    </t:EXCL>
</BODY>
</HTML>

```

onrowenter erzeugt wenn neue Daten verfügbar sind in der aktuellen Spalte
aufgrund Änderung der Daten in der Datenquelle zu Spalte

onrowexit erzeugt direkt Spaltenwechsel, also vor dem Verlassen der aktuellen Spalte
nur für databound-Objekte, die selbst Daten halten (Quelle und Ziel identisch)

onrowsdelete erzeugt direkt vor dem Löschen von Spalten

onrowsinserted erzeugt direkt nach dem Einfügen einer Spalte per Methode .AddNew()

onsave erzeugt, wenn Webseite als Datei gespeichert wird
oder Webseite als Bookmark in die Favoritenliste eingetragen wird
oder Webseite verlassen wird

Beispiel

```

<HTML>
<HEAD>
<META NAME="save" CONTENT="favorite">
<STYLE>
    .saveFavorite { behavior:url(#default#savefavorite); }
</STYLE>
</HEAD>
<BODY>
    <INPUT TYPE=text ID="ID_Input" CLASS=saveFavorite
        onsave="javascript:alert('Event onsave erkannt');"
    >
</BODY>
</HTML>

```

onscroll erzeugt wenn User die Scrollpfeile benutzt (nicht den Scrollbalken)
Syntax:

```

HTML      <ELEMENT onscroll = "handler" ... >
Script    object.onscroll = handler
           <SCRIPT FOR = object EVENT = onscroll>

```

onseek erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:

```

    .seekActiveTime()
    .seekSegmentTime()
    .seekTo()
    .seekToFrame()

```

siehe Objekt currTimeState und Behavior .style.time2

Beispiel :

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    var FrameAnzahl=600;

    function Suchen()
    {
        // prüfen ob Element nicht aktiv ist
        if (!ID_Video.currTimeState.isActive)
        {
            // nicht aktiv, also starten
            ID_Video.beginElement();
        }
        else
        {
            // prüfen des Eingabewertes
            if ( ( isFinite(ID_Input.value) )
                && (ID_Input.value < FrameAnzahl )
                && (ID_Input.value > 0)
            )
            {
                // ist okay, also ab Zeitpunkt animieren
                ID_Video.seekToFrame(ID_Input.value);
            }
            else
            {
                // fehlerhaft
                alert( "Fehler ! Frame-Wert muss > 0 und < "

```



```

        + FrameAnzahl
        + " sein"
    );
    ID_Input.focus();
}
}
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Video.currentFrame);"
    >
        0
    </SPAN>
    <BR>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        STYLE="width:175px; height:150px;"
        onmediacomplete="ID_Span.innerText= ID_Video.mediaDur;"
        onseek="alert('Der Frame ' + ID_Input.value + ' wird eingestellt !')
    >
    </t:VIDEO>
    <BR>
    Dauer des AVI:
    <SPAN ID="ID_Span"></SPAN>
    &nbsp;&nbsp;&nbsp;Sekunden
    <BR>
    setze seekToFrame:
    <INPUT TYPE="text"
        VALUE=""
        NAME="ID_Input"
        SIZE="3"
    >&nbsp;&nbsp;&nbsp;Sekunden
    <BR>
    <BUTTON onclick="Suchen();">
        Klick fuer Seek
    </BUTTON>
    <BUTTON onclick=" ID_Video.beginElement()">
        Restart
    </BUTTON>
</BODY>
</HTML>

```

onselect erzeugt wenn etwas selektiert wird

onselectionchange erzeugt wenn Selektionsstatus im Dokument verändert wird

onselectstart erzeugt wenn Objekt selektiert wird

onshow erzeugt wenn der Media Bar Player gerade sichtbar gemacht wird (nicht versteckt wird)
siehe Eigenschaft .enabled

entspricht dem Öffnen des Media Bar Player durch den User

Media Bar Player ist der Windows Media Player

siehe Behavior .style.mediaBar

onstart erzeugt mit Beginn eines jedem Loop-Durchlaufes des des Marquee-Objektes

Anzahl der Durchläufe laut LOOP-Attribut

LOOP-Attribut muss Wert > 0 haben, wenn onstart erzeugt werden soll

Beispiel

```

<BODY>
    <MARQUEE BEHAVIOR="alernate" LOOP=2 onstart="alert('onstart-Ereignis erkannt')">
        Marquee Text
    </MARQUEE>
</BODY>

```

onstop erzeugt wenn STOP-Button im Browser-Menü gedrückt wurde

wird direkt nach den onbeforeunload Event erzeugt

wird vor dem onunload Event erzeugt, da die Webseite verlassen wird

Beispiel

```

var TimerID;

function Rekursion()
{
    // nach belieben etwas tun
}

```



```
function TimerLoeschen()
{window.clearInterval(TimerID); }

function Init()
{TimerID = window.setInterval("Rekursion()",1000); }

document.onstop= TimerLoeschen;      // ohne ()
window.onload=Init;                  // ohne ()
```

onsubmit erzeugt wenn das Formular gesendet wird
 Eventhandler **muss** einen Rückkehrcode liefern (true oder false) z.B. per return-Anweisung
 Sendeaktion erzeugbar per

 Art der Formularaktion laut ACTION-Attribut des Formular-Tag
 Senden unterbindbar, wenn Eventhandler false liefert als Rückkehrcode (muss programmiert werden)
 z.B. wenn Daten im Formular fehlerhaft vom User eingegeben wurden

Beispiel

```
<BODY>
<FORM NAME="Formular" onsubmit="return(SubmitEventHandler());"></FORM>
</BODY>
```

onsyncrestored erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird
 nach vorausgegangenem Abbruch der Synchronisation
 siehe onoutofsync
 sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked"
 siehe Objekt currTimeState und Behavior .style.time2

ontimeerror erzeugt bei Zeitfehler nur für BODY-Objekt bei benutzung der Timeline

Beispiel

```
<SCRIPT FOR="BODY" event="ontimeerror">
alert("HTML+TIME error erkannt");
</SCRIPT>
```

ontrackchange erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx)
 in der Playliste gewechselt wurde

Beispiel für Aufbau einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
<ENTRY>
<TITLE>First title</TITLE>
<AUTHOR>Test</AUTHOR>
<COPYRIGHT>2002</COPYRIGHT>
<ABSTRACT>WAV File</ABSTRACT>
<REF href=""></REF>
<banner href = "first_title.gif">
<moreinfo href = "first_title.doc"><moreinfo>
<abstract>Visit the first abstract Web site</abstract>
</banner>
</ENTRY>
</ASX>
```

onunload siehe Objekt currTimeState und Behavior .style.time2
 erzeugt mit dem direkt vor dem Unload eines Dokumentes
 unload = verlassen des Dokumentes durch Laden eines neuen Dokumentes in das selbe Fenster

Beispiel

```
<HEAD>
<SCRIPT FOR=window EVENT=onunload>
alert("Event onunload erkannt");
</SCRIPT>

<SCRIPT>
function Umlenken()
{location.href="test.htm";}
</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE=button VALUE="Klicken ... weiter zu TEST.HTM" onclick="Umlenken()">
<IMG SRC="test.gif">
</BODY>
```

onURLFlip erzeugt wenn ein Scriptkommando, das in einer Advanced Streaming Format (ASF)-Datei (*.asf)
 liegen, ausgeführt wird
 siehe Objekt currTimeState und Behavior .style.time2
 Hinweis: window.event.URL Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-
 Datei von Element auf der Timeline
 es muss Ereignis onURLFlip aufgetreten sein
 siehe Objekt currTimeState und Behavior .style.time2

4.3.2.2.5.4.6. Eventarten wichtiger Objekte (Auswahl)



a Objekt

onactivate	ondragend	onmousemove
onafterupdate	ondragenter	onmouseout
onbeforeactivate	ondragleave	onmouseover
onbeforecopy	ondragover	onmouseup
onbeforecut	ondragstart	onmousewheel
onbeforedeactivate	ondrop	onmove
onbeforeeditfocus	onerrorupdate	onmoveend
onbeforepaste	onfocus	onmovestart
onbeforeupdate	onfocusin	onpaste
onblur	onfocusout	onpropertychange
onclick	onhelp	onreadystatechange
oncontextmenu	onkeydown	onresize
oncontrolselect	onkeypress	onresizeend
oncopy	onkeyup	onresizestart
oncut	onlosecapture	onselectstart
ondblclick	onmousedown	ontimeerror
ondeactivate	onmouseenter	
ondrag	onmouseleave	

applet Objekt

onactivate	ondeactivate	onmouseup
onbeforeactivate	onfocus	onmousewheel
onbeforecut	onfocusin	onmove
onbeforedeactivate	onfocusout	onmoveend
onbeforeeditfocus	onhelp	onmovestart
onbeforepaste	onkeydown	onpaste
onblur	onkeypress	onpropertychange
oncellchange	onkeyup	onreadystatechange
onclick	onload	onresize
oncontextmenu	onlosecapture	onresizeend
oncontrolselect	onmousedown	onresizestart
oncut	onmouseenter	onrowenter
ondataavailable	onmouseleave	onrowexit
ondatasetchanged	onmousemove	onrowsdelete
ondatasetcomplete	onmouseout	onrowsinserted
ondblclick	onmouseover	onscroll

area Objekt

onactivate	ondragend	onmouseleave
onbeforeactivate	ondragenter	onmousemove
onbeforecopy	ondragleave	onmouseout
onbeforecut	ondragover	onmouseover
onbeforedeactivate	ondragstart	onmouseup
onbeforeeditfocus	ondrop	onmousewheel
onbeforepaste	onfocus	onmove
onblur	onfocusin	onmoveend
onclick	onfocusout	onmovestart
oncontextmenu	onhelp	onpaste
oncontrolselect	onkeydown	onpropertychange
oncopy	onkeypress	onreadystatechange
oncut	onkeyup	onresizeend
ondblclick	onlosecapture	onresizestart
ondeactivate	onmousedown	onselectstart
ondrag	onmouseenter	ontimeerror

bgsound Objekt

onlayoutcomplete	onmouseleave
onmouseenter	onreadystatechange

body Objekt

onactivate	oncontrolselect	onfilterchange
onafterprint	oncut	onfocusin
onbeforeactivate	ondblclick	onfocusout
onbeforecut	ondeactivate	onkeydown
onbeforedeactivate	ondrag	onkeypress
onbeforeeditfocus	ondragend	onkeyup
onbeforepaste	ondragenter	onload
onbeforeprint	ondragleave	onlosecapture
onbeforeunload	ondragover	onmousedown
onclick	ondragstart	onmouseenter
oncontextmenu	ondrop	onmouseleave



onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove

onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend

onresizestart
onscroll
onselect
onselectstart
onunload

button Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondragenter

ondragleave
ondragover
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

comment Objekt

onpropertychange

onreadystatechange

custom Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart

div Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlayoutcomplete
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart
ontimeerror

document Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate

onbeforeeditfocus
onbeforepaste
onclick
oncontextmenu

oncontrolselect
oncut
ondblclick
ondeactivate



ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onmousedown
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove

onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectionchange
onstop

embed Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate

onfocus
onfocusin
onfocusout
onhelp
onload
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll

fieldset Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

font Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlayoutcomplete
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

form Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect

oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop

onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave



onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove

onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onreset

onresize
onresizeend
onresizestart
onselectstart
onsubmit
ontimeerror

frame Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeupdate
onblur
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onload
onmove
onmoveend

onmovestart
onresize
onresizeend
onresizestart

frameset Objekt

onactivate
onafterprint
onbeforedeactivate
onbeforeprint
onbeforeunload
onblur

oncontrolselect
ondeactivate
onfocus
onload
onmove
onmoveend

onmovestart
onresizeend
onresizestart
onunload

head Objekt

onlayoutcomplete

onreadystatechange

html Objekt

onlayoutcomplete
onmouseenter

onmouseleave
onreadystatechange

html comment Objekt

onlayoutcomplete

iframe Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeupdate
onblur
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onload
onmove
onmoveend

onmovestart
onreadystatechange
onresizeend
onresizestart
ontimeerror

img Objekt

onabort
onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerror
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onload
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input Objekt

keine Events

input button Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu

oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover

ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress



onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste

onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input checkbox Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

input file Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input hidden Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeeditfocus
onbeforeupdate
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onlosecapture
onmove
onmoveend

onmovestart
onpropertychange
onreadystatechange
onresizeend
onresizestart
ontimeerror

input image Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input password Objekt

onactivate
onafterupdate

onbeforeactivate
onbeforecut

onbeforedeactivate
onbeforeeditfocus



onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart

ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input radio Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

input reset Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input submit Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ontimeerror

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart

input text Objekt

onactivate

onafterupdate

onbeforeactivate



onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onchange
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave

ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselect
onselectstart
ontimeerror

label Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart

link Objekt

onload

onreadystatechange

map Objekt

onbeforeactivate
onbeforecut
onbeforepaste
onclick
oncut
ondblclick
ondrag
ondragend
ondragenter
ondragleave
ondragover

ondragstart
ondrop
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onpaste
onpropertychange
onreadystatechange
onscroll
onselectstart

marquee Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onbounce
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfinish
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart
onstart
ontimeerror



namespace Objekt

onreadystatechange

noframe Objekt

onreadystatechange

noscript Objekt

onreadystatechange

object Objekt

onactivate
 onbeforedeactivate
 onbeforeeditfocus
 onblur
 oncellchange
 onclick
 oncontrolselect
 ondataavailable
 ondatasetchanged
 ondatasetcomplete
 ondblclick
 ondeactivate
 ondrag

ondragend
 ondragenter
 ondragleave
 ondragover
 ondragstart
 ondrop
 onerror
 onfocus
 onkeydown
 onkeypress
 onkeyup
 onlosecapture
 onmove

onmoveend
 onmovestart
 onpropertychange
 onreadystatechange
 onresize
 onresizeend
 onresizestart
 onrowenter
 onrowexit
 onrowsdelete
 onrowsinserted
 onscroll
 onselectstart

option Objekt

onlayoutcomplete
 onlosecapture

onpropertychange
 onreadystatechange

onselectstart
 ontimeerror

popup Objekt

keine Events

scriptObjekt

onload

onpropertychange

onreadystatechange

select Objekt

onactivate
 onafterupdate
 onbeforeactivate
 onbeforecut
 onbeforedeactivate
 onbeforeeditfocus
 onbeforepaste
 onbeforeupdate
 onblur
 onchange
 onclick
 oncontextmenu
 oncontrolselect
 oncut
 ondblclick
 ondeactivate

ondragenter
 ondragleave
 ondragover
 ondrop
 onerrorupdate
 onfocus
 onfocusin
 onfocusout
 onhelp
 onkeydown
 onkeypress
 onkeyup
 onlosecapture
 onmousedown
 onmouseenter
 onmouseleave

onmousemove
 onmouseout
 onmouseover
 onmouseup
 onmousewheel
 onmove
 onmoveend
 onmovestart
 onpaste
 onpropertychange
 onreadystatechange
 onresize
 onresizeend
 onresizestart
 onselectstart

selection Objekt

ontimeerror

span Objekt

onactivate
 onafterupdate
 onbeforeactivate
 onbeforecopy
 onbeforecut
 onbeforedeactivate
 onbeforeeditfocus
 onbeforepaste
 onbeforeupdate
 onblur
 onclick
 oncontextmenu
 oncontrolselect
 oncopy
 oncut
 ondblclick
 ondeactivate
 ondrag

ondragend
 ondragenter
 ondragleave
 ondragover
 ondragstart
 ondrop
 onerrorupdate
 onfilterchange
 onfocus
 onfocusin
 onfocusout
 onhelp
 onkeydown
 onkeypress
 onkeyup
 onlosecapture
 onmousedown
 onmouseenter
 onmouseleave

onmouseleave
 onmousemove
 onmouseout
 onmouseover
 onmouseup
 onmousewheel
 onmove
 onmoveend
 onmovestart
 onpaste
 onpropertychange
 onreadystatechange
 onresize
 onresizeend
 onresizestart
 onselectstart
 ontimeerror



style Objekt

onerror	onreadystatechange
---------	--------------------

table Objekt

onactivate	ondragover	onmouseover
onbeforeactivate	ondragstart	onmouseup
onbeforecut	ondrop	onmousewheel
onbeforedeactivate	onfilterchange	onmove
onbeforeeditfocus	onfocus	onmoveend
onbeforepaste	onfocusin	onmovestart
onblur	onfocusout	onpaste
onclick	onhelp	onpropertychange
oncontextmenu	onkeydown	onreadystatechange
oncontrolselect	onkeypress	onresize
oncut	onkeyup	onresizeend
ondblclick	onlosecapture	onresizestart
ondeactivate	onmousedown	onscroll
ondrag	onmouseenter	onselectstart
ondragend	onmouseleave	ontimeerror
ondragenter	onmousemove	
ondragleave	onmouseout	

table.caption Objekt

onactivate	ondragenter	onmousemove
onbeforeactivate	ondragleave	onmouseout
onbeforecopy	ondragover	onmouseover
onbeforecut	ondragstart	onmouseup
onbeforedeactivate	ondrop	onmousewheel
onbeforepaste	onfocus	onmove
onblur	onfocusin	onmoveend
onclick	onfocusout	onmovestart
oncontextmenu	onhelp	onpaste
oncontrolselect	onkeydown	onpropertychange
oncopy	onkeypress	onreadystatechange
oncut	onkeyup	onresizeend
ondblclick	onlosecapture	onresizestart
ondeactivate	onmousedown	onselectstart
ondrag	onmouseenter	ontimeerror
ondragend	onmouseleave	

table.col Objekt

keine

table.colGroup Objekt

onreadystatechange

table.tBody Objekt

onactivate	ondragleave	onmousemove
onbeforeactivate	ondragover	onmouseout
onbeforecut	ondragstart	onmouseover
onbeforedeactivate	ondrop	onmouseup
onbeforepaste	onfocus	onmousewheel
onblur	onfocusin	onmove
onclick	onfocusout	onmoveend
oncontextmenu	onhelp	onmovestart
oncontrolselect	onkeydown	onpaste
oncut	onkeypress	onpropertychange
ondblclick	onkeyup	onreadystatechange
ondeactivate	onlosecapture	onresizeend
ondrag	onmousedown	onresizestart
ondragend	onmouseenter	onselectstart
ondragenter	onmouseleave	ontimeerror

table.tFoot Objekt

onactivate	oncut	onkeydown
onbeforeactivate	ondblclick	onkeypress
onbeforecut	ondeactivate	onkeyup
onbeforedeactivate	ondragenter	onlosecapture
onbeforepaste	ondragstart	onmousedown
onblur	onfocus	onmouseenter
onclick	onfocusin	onmouseleave
oncontextmenu	onfocusout	onmousemove
oncontrolselect	onhelp	onmouseout



onmouseover	onmovestart
onmouseup	onpaste
onmousewheel	onpropertychange
onmove	onreadystatechange
onmoveend	onresizeend

onresizestart
onselectstart
ontimeerror

table.tHead Objekt

onactivate	onfocus
onbeforeactivate	onfocusin
onbeforecut	onfocusout
onbeforedeactivate	onhelp
onbeforepaste	onkeydown
onblur	onkeypress
onclick	onkeyup
oncontextmenu	onlosecapture
oncontrolselect	onmousedown
oncut	onmouseenter
ondblclick	onmouseleave
ondeactivate	onmousemove
ondragenter	onmouseout
ondragstart	onmouseover

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tr Objekt

onactivate	ondragenter
onbeforeactivate	ondragleave
onbeforecopy	ondragover
onbeforecut	ondragstart
onbeforedeactivate	ondrop
onbeforeeditfocus	onfilterchange
onbeforepaste	onfocus
onblur	onfocusin
onclick	onfocusout
oncontextmenu	onhelp
oncontrolselect	onkeydown
oncopy	onkeypress
oncut	onkeyup
ondblclick	onlosecapture
ondeactivate	onmousedown
ondrag	onmouseenter
ondragend	onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tr.td Objekt

onactivate	ondragenter
onbeforeactivate	ondragleave
onbeforecopy	ondragover
onbeforecut	ondragstart
onbeforedeactivate	ondrop
onbeforeeditfocus	onfilterchange
onbeforepaste	onfocus
onblur	onfocusin
onclick	onfocusout
oncontextmenu	onhelp
oncontrolselect	onkeydown
oncopy	onkeypress
oncut	onkeyup
ondblclick	onlosecapture
ondeactivate	onmousedown
ondrag	onmouseenter
ondragend	onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tr.th Objekt

onactivate	ondeactivate
onbeforeactivate	ondragenter
onbeforecopy	ondragstart
onbeforecut	onfilterchange
onbeforedeactivate	onfocus
onbeforepaste	onfocusin
onblur	onfocusout
onclick	onhelp
oncontextmenu	onkeydown
oncontrolselect	onkeypress
oncopy	onkeyup
oncut	onlosecapture
ondblclick	onmousedown

onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange



onresizeend	onselectstart
onresizestart	ontimeerror

textarea Objekt

onactivate	ondragenter	onmouseout
onafterupdate	ondragleave	onmouseover
onbeforeactivate	ondragover	onmouseup
onbeforecopy	ondragstart	onmousewheel
onbeforecut	ondrop	onmove
onbeforedeactivate	onerrorupdate	onmoveend
onbeforeeditfocus	onfilterchange	onmovestart
onbeforepaste	onfocus	onpaste
onbeforeupdate	onfocusin	onpropertychange
onblur	onfocusout	onreadystatechange
onchange	onhelp	onresize
onclick	onkeydown	onresizeend
oncontextmenu	onkeypress	onresizestart
oncontrolselect	onkeyup	onscroll
oncut	onlosecapture	onselect
ondblclick	onmousedown	onselectstart
ondeactivate	onmouseenter	ontimeerror
ondrag	onmouseleave	
ondragend	onmousemove	

textrange Objekt

keine Events

var Objekt

onactivate	ondragleave	onmouseout
onbeforeactivate	ondragover	onmouseover
onbeforecut	ondragstart	onmouseup
onbeforedeactivate	ondrop	onmousewheel
onbeforeeditfocus	onfocus	onmove
onbeforepaste	onfocusin	onmoveend
onblur	onfocusout	onmovestart
onclick	onhelp	onpaste
oncontextmenu	onkeydown	onpropertychange
oncontrolselect	onkeypress	onreadystatechange
oncut	onkeyup	onresize
ondblclick	onlosecapture	onresizeend
ondeactivate	onmousedown	onresizestart
ondrag	onmouseenter	onselectstart
ondragend	onmouseleave	ontimeerror
ondragenter	onmousemove	

window Objekt

onactivate	ondeactivate	onmovestart
onafterprint	onerror	onresize
onbeforedeactivate	onfocus	onresizeend
onbeforeprint	onhelp	onresizestart
onbeforeunload	onload	onscroll
onblur	onmove	onunload
oncontrolselect	onmoveend	

xml Objekt

ondataavailable	onreadystatechange	onrowsdelete
ondatasetchanged	onrowenter	onrowsinserted
ondatasetcomplete	onrowexit	

4.3.2.2.5.5. Event-Behandlung beim Internet Explorer bzw. Netscape**4.3.2.2.5.5.1. Ansatz**

Zwischen Internet Explorer und Netscape bestehen prinzipielle Differenzen der Ereignisbehandlung, so dass Ereignisse z.B. onMouseOver verschiedenartig behandelt werden MÜSSEN. Das bedarf eines enormen Programmierungsaufwandes.

Internet Explorer arbeitet standardgemäß immer mit Eventbubbling:

Ereignis der aktuellen Stufe wird in die nächst höhere Stufe solange weitergereicht, bis das Ereignis verarbeitet wurde. Diese Vorgehensweise ist streng objektorientiert. Vorteil: Spätestens der Browser selber, kann auf das Ereignis reagieren. Selbstverständlich ist ebenfalls Eventcapturing möglich (Abfangen von Ereignissen).

Netscape arbeitet mit Eventcapturing:

Ereignis wird standardgemäß immer in höchster Ebene, also dem Browser registriert. Diese Vorgehensweise ist nicht streng objektorientiert, aber aufgrund der Tatsache nötig, dass der Netscape im Gegensatz zum Internet Explorer nicht standardgemäß über integrierte Komponenten des Betriebssystems verfügt.



Eventcapturing kann per Programmierung geändert werden:

Sollte die aktuelle Ebenen das Ereignis nicht verarbeiten können, so kann das Ereignis solange in die nächst tiefere Ebene durchgereicht werden, bis das Ereignis auch verarbeitet wurde. Nachteile sind: Die letzte Ebene muss unbedingt das Ereignis an das window.objekt weiterleiten, wenn diese letzte Ebene das Ereignis nicht verarbeiten kann. Es besteht reichlich Programmierungsaufwand
Der Eventhandler muss dem Ereignis als Parameter übergeben bekommen.

4.3.2.2.5.5.2. Ereignis und Eventhandler

Die Implementation der Events unterscheiden sich nicht nur zwischen den Browsern Internet Explorer und Netscape sondern auch innerhalb deren Versionen. Es ist daher immer zu prüfen, ob das Event überhaupt und dann auch mit den Eigenschaften implementiert ist:

bei IE	<pre>if (document.event) { if (.document.event.eigenschaft) }</pre>
	<p>für event ist das das konkrete Event einzutragen z.B. document.onmouseover für eigenschaft ist eine konkrete zu kodieren z.B. if (document.onmouseover.button)</p>
bei NS	nicht möglich, da kein direkter Zugriff auf das Objekt Event erlaubt ist

Es ist zu beachten, dass beim Netscape im Gegensatz zum Internet Explorer das Ereignis nicht als Objekt direkt angesprochen werden darf, sondern immer nur über den Parameter zum Eventhandler. Der Bezeichner des Paramaters ist frei wählbar. NS weiss, dass dieser Parameter das Eventobjekt referenziert. Es ist also auch der Bezeichner "event" wählbar, der beim Internet Explorer allerdings das Event-Objekt als vordefinierten Bezeichner darstellt. Damit ist klar, dass Bezeichner "event" nur in Eventhandlern des Netscape kodiert werden kann. Für Eventhandler, die IE **und** NS verwalten sollen, darf der Bezeichner nicht "event" lauten !!!

Ein Eventhandler für den Internet Explorer UND den Netscape muss prinzipiell wie folgt aufgebaut sein:

```
// Browser feststellen
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

function EvenhandlerFuerIEundNS(Ereignis_Paramater_der_nur_vom_NS_benoetigt_wird)
{
    if (ie)
    {
        // hier den Parameter IE-gerecht füllen
        Ereignis_Paramater_der_nur_vom_NS_benoetigt_wird = event;
    }
    // hier die Aktionen kodieren, wobei nun NS und IE den Parameter nutzen können
    // beachte: NS und IE haben erheblich unterschiedliche Event-Eigenschaften
    // so dass wieder mit if (ns) bzw. if (ie) gearbeitet werden muss !!

    if (ns)
    {
        // für den NS muss das Abfangen des Ereignisses XXX anstelle des Abfangens durch den Browser erst aktiviert
        // werden
        document.captureEvents(Event.XXX);
    }

    var RetteOnxxx= onxxx;                                     // retten nicht vergessen !!
    document.onxxx = EvenhandlerFuerIEundNS;                  // immer OHNE () kodieren !!!!, sonst
                                                                // wird kein Zeiger übergeben werden !!!

    Hinweis: XXX bzw.xxx ist das Event
            XXX beim NS:    vordefiniertes Ereignis-Schlüsselwörter
                           z.B. ONCLICK
            xxx             z.B. click
```

In einem Eventhandler für Tastatur- und Mausereignisse sollte keine alert()-Anweisung kodiert werden, da sonst sich der Browser aufhängen kann !!! Dafür die Statuszeile des Browsers benutzen: window.status = '.....';

Beim Internet Explorer muss das Objekt event ausgewertet werden, also welche Art von Ereignis das Objekt event gerade anbietet. Das wird im Eventhandler getan.

Bei Netscape muss das Ereignis als Parameter dem Eventhandlers übergeben werden.

Bestimmte Eventhandler können nachfolgende Ereignisse oder Aktionen ein- oder abschalten. Dazu muss der Eventhandler entweder return true; oder return false; besitzen. Alternativ wäre nur beim IE auch event.returnValue = true; bzw. event.returnValue = false; möglich.

Beispiel für einen Eventhandler für das Event onmouseover

(TWS) Microsoft JScript für den Hobby-Programmierer



```

function EventHandlerRoutineFuerOnMouseOver(Ereignis)
{
    if (document.all)
    {
        // IE
        // aktuelles Ereignis zuweisen, um es auswerten zu können
        Ereignis = event;    // Ereignis wird damit zum Zeiger !!

        // Netscape erhält sein Ereignis über den Parameter, der
        // immer ein Zeiger auf das Eventobjekt ist
    }

    // hier die Reaktion auf das Ereignis onmouseover kodieren
    // also die Eigenschaft von Ereignis anwenden

    // nur beim NS muss das Abfangen des Ereignisses aktiviert werden
    if (document.layers)
    {
        // NS
        // Abfangen des Ereignisses für den Eventhandler ermöglichen,
        // da standardgemäß das Ereignis vom
        // window-Objekt bearbeitet wird
        document.captureEvents(Event.MOUSEOVER);
    }
}
// Eventhandler dem Ereignis onmouseover zuweisen
document.onmouseover = EventHandlerRoutineFuerOnMouseOver;

```

4.3.2.2.5.5.3. Eventbehandlung im Netscape

Die Standard-Eventbehandlung wird beim Netscape immer vom window-Objekt selbst realisiert. Damit landen alle Events immer beim obersten Objekt. Eine davon abweichende Eventbehandlung ist zu programmieren.

4.3.2.2.5.5.3.1. Event des Netscape einer Nicht-Standardbehandlung unterziehen

4.3.2.2.5.5.3.1.1. Event und Eventhandler dem Objekt window zuordnen (captureEvents(event_liste))

Diese Methode unterbindet die Ereignisregistrierung des Browsers. Damit MUSS das Ereignis von einem Eventhandler verarbeitet werden. Der Eventhandler kann mehrere Ereignisse verarbeiten.

```
window.captureEvents(Event.event_schluessselwort_liste);
```

event_schluessselwort_liste: Folge von Eventbezeichnern mit Trennung durch | also logisches Oder

Bsp. für Eventbezeichner

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS
onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER
onkeydown	Event.KEYDOWN
onkeypress	Event.KEYPRESS
onkeyup	Event.KEYUP
onresize	Event.RESIZE

```

function MouseDownHandler(Ereignis)
{ .. }

```

```
window.captureEvents(Event.MOUSEDOWN);
```

```
window.onmouseover=MouseDownHandler;    // Achtung: nicht () kodieren !!
```

Wenn das Ereignis nur einmal bearbeitet und danach wieder der Standardbehandlung zugeordnet werden soll, so muss die Funktion am Ende .releaseEvents() zum Ereignis aufrufen.

Wenn das Ereignis nicht durch nachgelagerte Eventhandler in der Handlerhierarchie bearbeitet werden soll, so muss die Funktion mit return false; enden (sonst return true;).

4.3.2.2.5.5.3.1.2. Event entlang der Eventhierarchie weiterreichen

Es besteht die Möglichkeit, dass ein privater Eventhandler für die Weitergabe des Events die Standardhierarchie benutzt oder einen ausgewählten Eventhandler aufruft.



4.3.2.2.5.3.1.2.1. Event entlang der Nicht-Standard- Eventhierarchie weiterreichen (handleEvent(event_objekt))

Diese Methode ruft den aktuell dem Ereignis laut Parameter event_objekt zugeordneten Eventhandler auf.

Diese Methode wird also innerhalb des Programmcodes eines Eventhandlers verwendet, der das Ereignis von einem anderen Eventhandler als Argument bekommen hat.

4.3.2.2.5.3.1.2.2. Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))

Diese Methode aktiviert das Weiterreichen des Events zum in der Eventhierarchie untergeordneten Eventhandler.

4.3.2.2.5.3.2. Event des Netscape von Nicht-Standardbehandlung wieder der Standardbehandlung unterziehen

4.3.2.2.5.3.2.1. Standard-Eventhandler dem Objekt window zuordnen (releaseEvents(event_liste))

Diese Methode aktiviert die Ereignisregistrierung durch den Browser, ist also der Gegensatz zu .captureEvents(). Es dürfen nur Ereignisse releast werden, die auch gecaptured wurden.

```
window.releaseEvents(Event.event_schluessselwort_liste);
```

event_schluessselwort_liste: Folge von Eventbezeichnern mit Trennung durch | also logisches Oder

Bsp. für Eventbezeichner

<u>onXXX</u>	<u>Event.XXX</u>
onblur	Event.BLUR
ondragdrop	Event.DRAGDROP
onerror	Event.ERROR
onfocus	Event.FOCUS
onload	Event.LOAD
onmove	Event.MOVE
onresize	Event.RESIZE
onunload	Event.UNLOAD
onmouseover	Event.MOUSEOVER
onkeydown	Event.KEYDOWN
onkeypress	Event.KEYPRESS
onkeyup	Event.KEYUP
onresize	Event.RESIZE

Beispiel:

```
function EinmaligerMouseDownHandler()
{
    .....
    window.releaseEvents(Event.MOUSEDOWN);
}
```

```
window.captureEvents(Event.MOUSEDOWN);
```

```
window.onmouseover=EinmaligerMouseDownHandler; // Achtung: nicht () kodieren !!
```

Wenn das Ereignis nicht durch nachgelagerte Eventhandler in der Handlerhierarchie bearbeitet werden soll, so muss die Funktion mit return false; enden (sonst return true;).

4.3.2.2.5.3.2.2. Event entlang der Standard-Eventhierarchie weiterreichen (routeEvent(ereignis_objekt))

Diese Methode aktiviert das Weiterreichen des Events zum in der Eventhierarchie untergeordneten Eventhandler.

4.3.2.2.5.3.3. Eventbehandlung durch eine Fremdseite mit signiertem Script

Eventbehandlung des Netscape für eine Seite mit Frame, in dem ein fremdes HTML-Dokument angezeigt wird:

Es ist möglich, dass die fremde Seite die Ereigniskontrolle derjenigen Seite übernimmt, die das Frame enthält, in dem die Fremdseite angezeigt wird. Dieser Trojanereffekt ist nur dann realisierbar, wenn die Fremdseite ein signiertes Script enthält: Dieses Script muss sich beim Eigentümer der Seite, die den Frame besitzt, das Recht (Privileg) zur Übernahme der Ereigniskontrolle beschaffen. Dazu wird der User, in dessen Browser die Seite mit dem Frame läuft, befragt. Der Netscape hat einen Privileg-Manager integriert.

Hinweis: Beim Internet Explorer wird nur ActiveX (Betriebssystem-Schnittstelle) verwendet. Privilegien und signierte Script gibt es nicht. ActiveX wird vorallem bei der Integration von Fremderweiterungen der Browserfähigkeiten (beim NS als Plugin bezeichnet) benutzt.

4.3.2.2.5.3.3.1. Beschaffung der Rechte "UniversalBrowserWrite" bzw. "UniversalBrowserWrite" per Privilegmanger

Die fremde Seite kann z.B. folgenden Code besitzen:

```
netscape.security.PrivilegeManager.enablePrivilege("UniversalBrowserWrite");
window.enableExternalCapture();
window.captureEvents(Event.CLICK | Event.MOUSEDOWN);
```

Bei Ausführung des Codes wird der Manager aktiv, um die Rechtegenehmigung durch den



User zu beschaffen, da dieser Code (Script) vom User signiert sein muss.
 Der Netscape-Privilegmanager sichert ab, dass der Browseruser gefragt wird, ob er eine Privilegänderung gegenüber den Standard-Privilegien dulden will.
 Für ein signiertes Script muss das Privileg "UniversalBrowserWrite" bzw. "UniversalBrowserWrite" vom Privilegmanager angefordert sein.

4.3.2.2.5.5.3.3.2. Eventbehandlung durch Fremde Seite aktivieren (enableExternalCapture())

Diese Methode aktiviert die Kontrolle einer Fremdseite, die in einem Frame dsrgestellt wird, auf die Seite, die den Frame inne hat. Die Ereignissüberwachung in einem Fenster wird somit aktiv.

Diese Methode benötigt ein signiertes Script.

Das Event muss mit captureEvents() dem Window-Objekt zugeordnet werden.

Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !

4.3.2.2.5.5.3.3.3. Eventbehandlung durch Fremde Seite deaktivieren (disableExternalCapture())

Diese Methode deaktiviert die Kontrolle einer Fremdseite, die in einem Frame dsrgestellt wird, auf die Seite, die den Frame inne hat. Die Ereignissüberwachung in einem Fenster wird somit aktiv.

Diese Methode benötigt ein signiertes Script.

Das Event muss mit captureEvents() dem Window-Objekt zugeordnet werden.

Achtung: Anzeige der fremden Seite in einem Frame ist rechtlich nur mit Einverständnis des Herstellers der Fremdseite zulässig !

4.3.2.2.5.5.3.4. Beispiel

Es ist zu beachten, dass beim Netscape im Gegensatz zum Internet Explorer das Ereignis nicht als Objekt direkt angesprochen werden darf, sondern immer nur über den Parameter zum Eventhandler. Der Bezeichner des Paramaters ist frei wählbar. NS weiss, dass dieser Parameter das Eventobjekt referenziert. Es ist also auch der Bezeichner "event" wählbar, der beim Internet Explorer allerdings das Event-Objekt als vordefinierten Bezeichner darstellt. Damit ist klar, dass Bezeichner "event" nur in Eventhandlern des Netscape kodiert werden kann. Für Eventhandler, die IE und NS verwalten sollen, darf der Bezeichner nicht "event" lauten !!!

Beim Internet Explorer muss das Ereignis im Eventhandler immer direkt über das Objekt event (Unterobjekt des Objektes window) behandelt werden. Daher ist für den Eventhandler kein Parameter nötig.

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript">
<!--
    function MouseDownFuerInputButton(Ereignis)
    {
    }

    function OnMouseDownEventWeiterreichen(Ereignis)
    {window.routeEvent(Ereignis);} // In der Hierarchie weiterreichen

    window.captureEvents(Event.MOUSEDOWN);

    window.onmousedown= OnMouseDownEventWeiterreichen;
                          // Achtung: ohne () kodieren!!

// -->
</SCRIPT>
</HEAD>
<BODY>
    <FORM>
        <INPUT TYPE=button
                VALUE="Weiterreichen"
                onmousedown= "MouseDownFuerInputButton()"
            >
    </FORM>
</BODY>
</HTML>
```

Ablauf: Head-Teil: Der Script-Teil im Head wird zuerst abgearbeitet.

Body-Teil:

Es wird auf das Input-Button gedrückt.
 Wegen captureEvents() wird das Mausereignis ZUERST von
 OnMouseDownEventWeiterreichen
 bearbeitet. Damit wird auf das Weiterleiten aktiviert.
 Das untergeordnete Element ist der Input-Button. Damit wird
 MouseDownFuerInputButton
 aktiviert.

Hinweis: Dieses Beispiel hinkt, da die Standardbehandlung letztendlich das gleiche bewirkt



aber auf anderen Wegen.

4.3.2.2.5.5.4. Eventbehandlung beim Internet Explorer

Es ist zu beachten, dass beim Netscape im Gegensatz zum Internet Explorer das Ereignis nicht als Objekt direkt angesprochen werden darf, sondern immer nur über den Parameter zum Eventhandler. Der Bezeichner des Parameters ist frei wählbar. NS weiss, dass dieser Parameter das Eventobjekt referenziert. Es ist also auch der Bezeichner "event" wählbar, der beim Internet Explorer allerdings das Event-Objekt als vordefinierten Bezeichner darstellt. Damit ist klar, dass Bezeichner "event" nur in Eventhandlern des Netscape kodiert werden kann. Für Eventhandler, die IE und NS verwalten sollen, darf der Bezeichner nicht "event" lauten !!!

Beim Internet Explorer muss das Ereignis im Eventhandler immer direkt über das Objekt event (Unterobjekt des Objektes window) behandelt werden. Daher ist für den Eventhandler kein Parameter nötig.

4.3.2.2.5.5.4.1. Event des IE einer Nicht-Standardbehandlung unterziehen

(attachEvent(event_objekt, funktions_referenz))

Diese Methode ordnet dem Ereignis-Objekt laut Parameter event_bezeichner diejenige Funktion zu, die das Event behandeln soll. Soll das Event nur einmalig behandelt werden, so ist in der Funktion die Eventbehandlung per .detachEvent() aufzuheben.

Bsp.:

```
function handler_funktion(){..}
attachEvent('onmouseover', handler_funktion);
```

4.3.2.2.5.5.4.2. Event von Nicht-Standardbehandlung wieder der Standardbehandlung

unterziehen (detachEvent(event_objekt))

Diese Methode ordnet dem Ereignis-Objekt laut Parameter event_bezeichner wieder die Standard-Eventverarbeitung zu und hebt somit attachEvent() auf.

4.3.2.2.5.5.4.3. Ereignisbehandlung für mouseover und mouseout ein-bzw. ausschalten

(.releaseCapture() und .setCapture())

ausschalten: `releaseCapture()`
einschalten: `setCapture()`

4.3.2.2.5.5.4.4. Event bei Behavior (Verhaltensweise)

siehe Objekt `element` für Eventsteuerung eines Behavior per *.htc-Datei
`.style.time2` für Standard-Behavior des IE

4.3.2.2.5.5.5. Fehlerbehandlung per onerror

4.3.2.2.5.5.5.1. onerror-Standard abschalten

```
<SCRIPT ...>
<!--
    var OnErrorRetten = window.onerror;
    window.onerror=null; // null ist Schlüsselwort
// -->
</SCRIPT>
```

4.3.2.2.5.5.5.2. onerror-Routine privater Art (eigene Fehlerbehandlung einrichten)

```
onerror                Ereignis ausgelöst, wenn
                        während Ladens eines HTML-Dokumentes ein Syntaxfehler auftritt
                        eines Bildes ein Fehler auftritt
                        Abarbeitung von Scripten ein Runtime-Error auftritt
                        sämtliche Fehlermeldungen unterbinden per
                        var RetteOnErrorHandler = window.onerror;
                        window.onerror = null;
                        Eventhandler muss wie folgt kodiert werden:
                        function freier_name_fuer_onerror_behandlung
                        ( error_erklaerung_string,
                          url_des_html_dokumentes_als_string,
                          zeilen_nr_des_errors_im_html_dokument
                        )
                        {
                            .....
                            return true;           // nur wenn true geliefert, dann
                                                    // wird die Browsereigenen
                                                    // onerror-Behandlung
                                                    // unterdrückt
                        }
                        pro Fehler ein Aufruf des Eventhandlers
                        --> Folge von Fehlern, also Folge von Aufrufen
                        Die Script-Debugger-Aufforderung ist nur in den Eigenschaften des
                        Browsers abschaltbar
```

Beispiel 1:

```
<SCRIPT ...>
<!--
    function eigenes_fehler_fenster(meldungs_text,url,zeilen_nr)
    {
        // Fehlermeldung bilden
```



```

var url_als_kette=url.toString();
var zeilen_nr_als_kette=zeilen_nr.toString();
var meldung_komplett=meldungs_text + url_als_kette + zeilen_nr_als_kette;

// Meldungsfenster
var fenster=window.open();
with (fenster.document)
{
    open("text/html"); // HTML-Dokument im Fenster erzeugen

    writeln("<HTML><HEAD><TITLE>Private Errormeldung</TITLE></HEAD>");
    writeln("<BODY><H1>Fehlermeldung</H1>");

    writeln(meldung_komplett);

    writeln("<FORM><INPUT TYPE=BUTTON VALUE='OK'
onClick='self.close()'>");
    );           // ist EINE Zeile

    // Button anklicken, damit das Meldungs-Fenster geschlossen wird
    close(); //HTML-Dokument schliessen
}
return true;           // muss true liefern für window.onerror
}

var RetteOnError=window.onerror;

window.onerror= eigenes_fehler_fenster; // NICHT onError kodieren !
// keine () kodieren, da sonst die Funktion
// sofort ausgeführt wird !!
// window.onerror verlangt die Zuweisung
// von true für Abschaltung des
// Standard-onerror

// -->
</SCRIPT>
...
<BODY>
...
</BODY>

```

Beispiel 2:

```

<SCRIPT>
function FehlerAufspueren (MeldungString, UrlString, ZielenNummerString)
{
    ID_Div.innerHTML ="<B>Fehler erkannt</B>";
    ID_Div.innerHTML+="Error: " + MeldungString      + "<BR>";
    ID_Div.innerHTML+="Line: " + ZielenNummerString + "<BR>";
    ID_Div.innerHTML+="URL: " + UrlString            + "<BR>";

    return false;
}

window.onerror=FehlerAufspueren; // ohne () kodieren
</SCRIPT>
<BODY>
<DIV ID="ID_Div"></DIV>
<BODY>

```

Beispiel 3 für Bild:

```

<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette;           // wird sofort geparkt, also IMG-Tag ausgeführt
                                     // ( anstelle von eval()
                                     // bzw. document.write() )

    ID_IMG.src="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

```



```

function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>

```

4.3.2.2.5.5.6. Eventbehandlung in einem Formular des IE und NS

Es wird das Click-Event abgefangen und mit einer Funktion bearbeitet.

Das Formular wird bei erkannter Veränderung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE ="Javascript">
<!--
    // Browser feststellen
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    if (ns)
    {
        // nur für NS: Eventart CLICK wird abgefangen
        window.captureEvents(Event.ONCLICK);
    }

    // soll das Fenster selbst den Click-Eventhandler 1 bekommen, so hier kodieren
    // window.onclick=click_auswertung_1;
    // WICHTIG ist, dass das Fenster auch das Event WEITERLEITET !! nach unten

    function click_auswertung_1(Ereignis)
    {
        // folgende alternative Varianten sind möglich

        // nur für NS:
        // in der Objekt-Hierarchie weiterleiten
        if (ns)
        {routeEvent(Ereignis);}

        // für NS und IE:
        // nichts tun
        return true;

        // im aktuellen Fenster und dessen Dokument das Formular anwählen
        // und dem Eventhandler des 2. Buttons das Ereignis übergeben,
        // wobei der Eventhandler des 2. Buttons dieses Ereignis verarbeiten muss
        self.document.logischer_form_name.logischer_button_name_2.handleEvent(Ereignis);

        // die Auswertung hier kodieren --> bitte kein alert(); !!!
        if (ie)
        {
            Ereignis=event.button;
            // hier das Maus-Ereignis allgemein erfasst
            //          0      keine Maustaste gedrückt
            //          1      linke Maustaste gedrückt
            //          2      rechte Maustaste gedrückt
            //          4      mittlere Maustaste gedrückt
            //          Kombination aus 1 bis 4 für Maustastenkombination
            //          z.B. 3 = linke UND rechte Maustaste gedrückt (1 + 2 = 3)
            //          7 = alle Maustasten gedrückt (1 +2 +3 + 4 = 7)
            //          nur Internet Explorer ab 4.x
            .....
        }
        else
        {
            if (ns)
            {.....}
        }
    }

    function click_auswertung_2(Ereignis)
    { // analog zu click_auswertung_1 }

```



```
// -->
</SCRIPT>
</HEAD>
<BODY ....>
    <FORM NAME="logischer_form_name">
        <INPUT          TYPE=button
                        NAME="logischer_button_name_1"
                        onclick="click_auswertung_1();"
        >
        <INPUT          TYPE=button
                        NAME="logischer_button_name_2"
                        onclick="click_auswertung_2();"
        >
    </FORM>
</BODY>
</HTML>
```

4.3.2.2.5.5.7. Eventbehandlung bei Drag & Drop

4.3.2.2.5.5.7.1. Eventbehandlung bei Drag & Drop eines HTML-Elementes beim Internet Explorer

IE ermöglicht Drag&Drop von HTML-Elementen

auch über Fenster

per Maus: linke Maustaste auf dem Element drücken und gedrückt lassen,
dann Element ziehen an gewünschte Position,
dann linke Maustaste loslassen

Das Quell- und Ziel-Element müssen identischer Art sein !

4.3.2.2.5.5.7.1.1. Eventarten für Drag & Drop

ondrag	Ereignis ausgelöst, immer wenn HTML-Element verschoben wird, also permanent solange gezogen wird nur verwenden für Erfassung der aktuellen Element-Position Eventhandler für Quellobjekt kodieren: nur auf Ereignis reagieren, solange der Bereich des Zielobjektes noch nicht erreicht wurde private Boolean-Variable verwenden: if (false) ... reaktion per dragenter auf true setzen per dragleave auf false initialisieren
ondragend	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn HTML-Element verschoben, eventuell abgelegt und somit Drag & Drop nun beendet werden kann aktivieren des Ablegen per dragover Eventhandler für Quellobjekt kodieren
ondragenter	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn der Bereich des Ziel-Elementes das 1. Mal betreten wurde Eventhandler für Zielobjekt kodieren: muss die private Boolean-Variable auf true setzen per drag ausgewertet (dort auf false prüfen) per dragleave auf false initialisieren
ondragleave	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn während des Ziehens die Maustaste losgelassen wurde, wobei somit Drag & Drop zugleich abgebrochen wurde Eventhandler für Quellobjekt kodieren: muss die private Boolean-Variable auf false initialisieren per drag ausgewertet (dort auf false prüfen) per dragenter auf true gesetzt
ondragover	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn der Bereich des Ziel-Elementes ab 2. Mal betreten wurde also permanent fortlaufend, solange Maus im Bereich des Zieles ist und nicht abgelegt wurde Eventhandler für Zielobjekt kodieren: muss event.returnValue=false; enthalten, wenn auch Ablegen im Zielobjekt gewollt ist, also der Einfüge-Cursor mit Pfeil und Pluszeichen angezeigt werden soll (standardgemäß ist keine Ablage möglich, also Cursor mit dem durchgestrichenen Kreis sichtbar !)
ondragstart	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn Maus auf Quellobjekt dauergedrückt ist, also die Verschiebung damit beginnen kann Eventhandler für Quellobjekt kodieren
ondrop	Tag alle Tags, in denen Text markierbar sind: <DIV>, , Ereignis ausgelöst, wenn Quellobjekt auf dem Bereich des Zielobjektes abgelegt wird, falls es per dragend zugelassen wird Eventhandler für Zielobjekt kodieren Tag alle Tags, in denen Text markierbar sind: <DIV>, ,



4.3.2.2.5.5.7.1.2. Beispiel für Drag & Drop

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--

// Eine Grafik per Drag & Drop in das INPUT-Element verschieben und danach einen Text zur Grafik
// in die INPUT-Textzeile ablegen

// Quellobjekt und Zielobjekt müssen gleicher Elementart sein !

var ZielObjektErreicht=false;
var TextZurGrafik="Tanzende Frau";    // Text der durch Drag & Drop der Grafik eingefügt wird

// ##### Funktionen für das Quellobjekt

function Drag_Start_Ereignis_Routine()
{
    window.status = 'ondragstart ausgelöst für '+ event.srcElement.name
                    + '. Verschieben ist soeben gestartet worden ! Bitte Grafik in das Input-Textfeld
                                                    verschieben !';
}

function Drag_Ereignis_Routine()
{
    if (!ZielObjektErreicht)
    {
        window.status = 'ondrag ausgelöst für '+ event.srcElement.name
                        + '. Es wird gezogen ! Maus ist an der Position X,Y = '
                        + event.x + ',' + event.y;
    }
}

function Drag_End_Ereignis_Routine()
{
    window.status = 'ondragend ausgelöst für '+ event.srcElement.name
                    + '. Verschieben ist soeben beendet worden !';
}

function Drag_Leave_Ereignis_Routine()
{
    window.status = 'ondragleave ausgelöst für '+ event.srcElement.name
                    + '. Während ziehen wurde Maus losgelassen !';

    ZielObjektErreicht=false;
}

// ##### Funktionen für das Zielobjekt

function Drag_Enter_Ereignis_Routine()
{
    window.status = 'ondragenter ausgelöst für Zielobjekt ! Bereich des Zielobjektes wurde soeben betreten !';
    ZielObjektErreicht=true;
}

function Drag_Over_Ereignis_Routine()
{
    window.status = 'ondragover ausgelöst für Zielobjekt ! Maus im Bereich des Zielobjektes !
                    Ablegen nun möglich !';

    event.returnValue=false; // Zeigt den Einfüge-Cursor an, also Pfeil mit Pluszeichen
}

function Drop_Ereignis_Routine()
{
    window.status = 'ondrop ausgelöst für Zielobjekt ! Text vom Quellobjekt wurde eben eingefügt !';
    document.all.ZielObjekt.value=TextZurGrafik;
}
-->
</SCRIPT>
</HEAD>
<BODY>
Bitte Grafik per Drag und Drop langsam in die Textzeile verschieben und dabei die Statuszeile beobachten !
<BR>

```



```

<IMG NAME="QuellObjekt" SRC="picture.gif" ALT="dein bild" HEIGHT=49 WIDTH=30
      ondragstart ="Drag_Start_Ereignis_Routine()"
      ondrag      ="Drag_Ereignis_Routine()"
      ondragend   ="Drag_End_Ereignis_Routine()"
      ondragleave  ="Drag_Leave_Ereignis_Routine()"
>
<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
<INPUT TYPE="text" NAME="ZielObjekt" VALUE=""
      ondragenter="Drag_Enter_Ereignis_Routine()"
      ondragover="Drag_Over_Ereignis_Routine()"
      ondrop="Drop_Ereignis_Routine()"
>
</BODY>
</HTML>

```

4.3.2.2.5.5.7.2. Eventbehandlung bei Drag & Drop von Dateien und Verknüpfungen beim Netscape ab 4.x

NS ermöglicht Drag&Drop von Dateien und Verknüpfungen

auch über Fenster

per Maus: linke Maustaste auf das Datei bzw. Verknüpfung niederdrücken

und gedrückt lassen,

dann Element ziehen an gewünschte Position

dann linke Maustaste loslassen

Eventart: dragdrop

Ereignis ausgelöst, wenn eine Datei oder Verknüpfung verschoben wurde

Tag <BODY>

4.3.2.2.5.5.8. Tastatur-Eventbehandlung des IE und NS

Die Programmierung der Tastatur-Events unterscheidet sich nicht nur zwischen den Browsern IE und Netscape, sondern auch noch innerhalb deren Versionsnummern. Dieser Umstand zeugt vom divergierenden Ansatz der Browserhersteller, welcher im krassen Widerspruch zu einer Programmierungsfreundlichkeit steht.

Die bessere Konformität zur konsequent-objektorientierten Umsetzung des HTML-Dokumentes, seiner Elemente und Ereignisse hat der Internet Explorer, der damit ein weiteres Spektrum an Möglichkeiten zur Programmierung unter Javascript bietet, obwohl Microsoft weiterhin Visual Basic-Script als Komponente des Windows Script Host (Betriebssystem-Komponente) favorisiert und eine Dokumentierung der Javascript-Komponente (inklusive Debugger-Freeware-Tool zum aktuellen IE) konsequent vernachlässigt.

4.3.2.2.5.5.8.1. Tastatur-Eventbehandlung beim Internet Explorer

4.3.2.2.5.5.8.1.1. Tastatur-Eventarten

keydown Ereignis ausgelöst nicht für alle Tastenarten (siehe weiter unten)
 liefert ASCII-Code der Taste nach Eigenschaft .keyCode
 nach String umwandeln per string_name=fromCharCode(..)
 erste Mal, wenn Taste gedrückt
 periodisch, wenn Dauerdruck der Taste
 --> siehe Eigenschaft .repeat
 Besonderheit bei Kombination Steuertaste und andere Taste
 z.B. Shift-Taste und B-Taste zugleich für Großbuchstabe B
 Ereignis wird pro Tastendruck aufgerufen, also
 1. Mal für Shift-Taste
 2. Mal für B-Taste
 ab IE 5.x: keydown-Ereignis ruft automatisch das keypress-Ereignis auf, aber
 wenn keydown-Handler return false; bzw. beim IE
 alternativ event.returnValue=false; liefert, so wird ein
 kodierter keypress-Handler nicht aktiviert !

Tag <BODY>, <DIV>, <FORM>, <INPUT>, <SELECT>, ,
 <TABLE>, <TD>, <TEXTAREA>, <TR>,
 alle Text-Tags wie z.B. ,

keypress wird unmittelbar nach dem keydown-Ereignis ausgelöst allerdings nur für
 Tastenarten
 ! @ # \$ % ^ & * () _ - + = [] { } , . / ? \ | ' " ` ~
 0 bis 9
 a bis z
 beim IE: egal ob Gross oder Klein, da immer der ASCII-Code des
 Grossbuchstaben geliefert wird

Enter
 Leertaste
 und dient dem Erkennen von Tastaturkombinationen,
 die nicht vom Ereignis keydown abgefangen werden
 der Aufruf des keypress-Handlers ist **nur** möglich,
 wenn der keydown-Handler



bzw. beim IE alternativ `return true;`
`event.returnValue=true;`
 liefert
 Tag `<BODY>`, `<DIV>`, `<FORM>`, `<INPUT>`, `<SELECT>`, ``,
`<TABLE>`, `<TD>`, `<TEXTAREA>`, `<TR>`,
 alle Text-Tags wie z.B. ``, ``
 keyup Ereignis ausgelöst, wenn gedrückte Taste jeder Art losgelassen wird
 wird automatisch nach Ereignis keypress ausgelöst
 Ereignis keypress automatisch nach Ereignis keydown ausgelöst
 Tag `<BODY>`, `<DIV>`, `<FORM>`, `<INPUT>`, `<SELECT>`, ``,
`<TABLE>`, `<TD>`, `<TEXTAREA>`, `<TR>`,
 alle Text-Tags wie z.B. ``, ``

4.3.2.2.5.5.8.1.2. Tastatur-Eigenschaft

4.3.2.2.5.5.8.1.2.1. Alle Tasten (.keyCode und .repeat)

.keyCode ASCII-Code der gedrückten Taste
 wenn 0 so keine Taste gedrückt
 Umwandlung in String per `String.fromCharCode(TastenKodeASCII)`
 .repeat true, so Taste im Dauerdruck, sonst false

4.3.2.2.5.5.8.1.2.2. Steuertasten Alt, Strg (Ctrl) und Umschalt (Shift) (.xxxKey und .xxxLeft)

.altKey true wenn ALT-Taste gedrückt, sonst false (egal ob linke oder rechte)
 .altLeft true wenn linke ALT-Taste gedrückt, sonst false
 Hinweis: rechte Alt-Taste (AltGr) ermitteln aus
`altKey AND NOT altLeft`

.ctrlKey true, so Strg-Taste gedrückt (egal ob linke oder rechte)
 .ctrlLeft true wenn linke Strg-Taste gedrückt, sonst false
 Hinweis: rechte Strg-Taste ermitteln aus
`ctrlKey AND NOT ctrlLeft`

.shiftKey true, so shift-Taste gedrückt (egal ob linke oder rechte)
 .shiftLeft true wenn linke shift-Taste gedrückt, sonst false
 Hinweis: rechte shift-Taste ermitteln aus
`shiftKey AND NOT shiftLeft`

4.3.2.2.5.5.8.1.2.3. Eventhandler-Eigenschaften (.returnValue)

.returnValue enthält den Boolean-Rückkercode des Eventhandlers
`event.returnValue=true;` ist identisch `return true;`
`event.returnValue=false;` ist identisch `return false;`
 Eventhändler muss liefern
 true, um eine Aktion aufgrund des Events zuzulassen
 false, um eine Aktion aufgrund des Events nicht zuzulassen
 Anwendung z.B. bei RESET und SUBMIT vom Formular

4.3.2.2.5.5.8.1.2.4. HTML-Element-Event-Eigenschaft (.srcElement)

.srcElement Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde
 entweder im HTML-Tag definieren per `onXXX = " "`
 oder Bezug auf den logischen Namen des HTML-Elementes
`if (logischer_name = event.srcElement)`

4.3.2.2.5.5.8.1.2.5. Eventart-Eigenschaft (.type)

.type enthält die Event-Art z.B. `abort`

4.3.2.2.5.5.8.1.3. Tastatur-Ereignis-Folge onkeydown und onkeypress

Standard: erst `onkeydown`
 dann `onkeypress`

modifiziert: wenn Eventhandler für `keydown` `return false;` oder `event.returnValue=false;`
 enthält, so wird der Eventhandler für `keypress` nicht aktiviert

4.3.2.2.5.5.8.1.3.1. Tastatur-Ereignis onkeydown

onkeydown ausgelöst bei den Tasten
 ab IE 4.x `! @ # $ % ^ & * () _ - + = < [] { } , . / ? \ | ' " ` ~`
 0 bis 9
 a bis z
 egal ob Gross oder Klein, da immer der ASCII-Code des Grossbuchstaben
 geliefert wird
 Cursor-Tasten
 Einfg
 Ende
 Entf
 ESC
 F1 bis F12



Leertaste
 Pos1
 Umschalt (Shift)
 Tab
 ab IE 5.x zusätzlich Bild auf und ab
 Rück
 Umschalt+Tab (Shift+Tab)

Die Enter-Taste wird hier nicht geliefert !

liefert den ASCII-Code in die Event-Eigenschaft

.keyCode ASCII-Code der gedrückten Taste
 wenn 0 so keine Taste gedrückt
 wobei bei Buchstaben nicht unterschieden wird, ob gross oder klein,
 da immer der ASCII-Code des Grossbuchstaben geliefert wird
 Umwandlung in String per String.fromCharCode(TastenKodeASCII)

liefert Dauerdruckzustand in die Eventeigenschaft

.repeat true, so Taste im Dauerdruck, sonst false

folgende Tasten werden auch vom Ereignis keypress geliefert:

ab IE 4.x ! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ | ' " ` ~
 0 bis 9
 a bis z
 egal ob Gross oder Klein, da immer der ASCII-Code des Grossbuchstaben
 geliefert wird
 Enter
 Leertaste

folgende Tasten werden nicht vom Ereignis onkeypress geliefert:

Cursor-Tasten
 Einfg
 Ende
 Entf
 ESC
 F1 bis F12
 Pos1
 Umschalt (Shift)
 Tab
 Bild auf und ab
 Rück
 Umschalt+Tab (Shift+Tab)

Zusatzanalyse möglich per Event-Eigenschaften

.altKey true wenn ALT-Taste gedrückt, sonst false (egal ob linke oder rechte)
 .altLeft true wenn linke ALT-Taste gedrückt, sonst false
 Hinweis: rechte Alt-Taste (AltGr) ermitteln aus
 altKey AND NOT altLeft
 .ctrlKey true, so Strg-Taste gedrückt (egal ob linke oder rechte)
 .ctrlLeft true wenn linke Strg-Taste gedrückt, sonst false
 Hinweis: rechte Strg-Taste ermitteln aus
 ctrlKey AND NOT ctrlLeft
 .shiftKey true, so shift-Taste gedrückt (egal ob linke oder rechte)
 .shiftLeft true wenn linke shift-Taste gedrückt, sonst false
 Hinweis: rechte shift-Taste ermitteln aus
 shiftKey AND NOT shiftLeft

Unterbindung des Aufrufes des keypress-Handlers:

keydown-Handler muss return false; bzw. event.returnValue=false; liefern

4.3.2.2.5.8.1.3.2. Tastatur-Ereignis onkeypress

wird unmittelbar nach dem onkeydown-Ereignis ausgelöst und dient dem Erkennen von Tastaturkombinationen,
 die nicht vom Ereignis keydown abgefangen werden

der Aufruf des keypress-Handlers ist **nur** möglich,

wenn der keydown-Handler return true; bzw. event.returnValue=true; liefert

wird ausgelöst bei den Tasten

ab IE 4.x ! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ | ' " ` ~
 0 bis 9
 a bis z
 egal ob Gross oder Klein, da immer der ASCII-Code des Grossbuchstaben
 geliefert wird
 Enter
 Leertaste



folgende Tasten sind hier nicht auslesbar (nur per keydown):

Cursor-Tasten
Einfg
Ende
Entf
ESC
F1 bis F12
Pos1
Umschalt (Shift)
Tab
Bild auf und ab
Rück
Umschalt+Tab (Shift+Tab)

bereits mit dem Ereignis keydown sind abfangbar:

.altKey	true wenn ALT-Taste gedrückt, sonst false (egal ob linke oder rechte)
.altLeft	true wenn linke ALT-Taste gedrückt, sonst false Hinweis: rechte Alt-Taste (AltGr) ermitteln aus altKey AND NOT altLeft
.ctrlKey	true, so Strg-Taste gedrückt (egal ob linke oder rechte)
.ctrlLeft	true wenn linke Strg-Taste gedrückt, sonst false Hinweis: rechte Strg-Taste ermitteln aus ctrlKey AND NOT ctrlLeft
.shiftKey	true, so shift-Taste gedrückt (egal ob linke oder rechte)
.shiftLeft	true wenn linke shift-Taste gedrückt, sonst false Hinweis: rechte shift-Taste ermitteln aus shiftKey AND NOT shiftLeft
.repeat	liefert Dauerdruckzustand in die Eventeigenschaft true, so Taste im Dauerdruck, sonst false

4.3.2.2.5.5.8.2. Tastatur-Eventbehandlung beim Netscape

Der Event-Handler muss als Parameter das gewünschte Event übergeben bekommen

4.3.2.2.5.5.8.2.1. Tastatur-Eventarten

onkeydown	Ereignis ausgelöst für alle Tastenarten liefert ASCII-Code der Taste nach Eigenschaft.which nach String umwandeln per string_name=fromCharCode(..) einmalig, also genau wenn Taste gedrückt wird Besonderheit bei Kombination Steuertaste und andere Taste z.B. Shift-Taste und B-Taste zugleich für Großbuchstabe B Ereignis wird pro Tastendruck aufgerufen, also 1. Mal für Shift-Taste 2. Mal für B-Taste ab NS 4.x: keydown-Ereignis ruft automatisch das keypress-Ereignis auf, aber wenn keydown-Handler return false; bzw. beim IE alternativ event.returnValue=false; liefert, so wird ein kodierter keypress-Handler nicht aktiviert !
onkeypress	Tag <A>, , <TEXTAREA> wird unmittelbar nach dem keydown-Ereignis ausgelöst allerdings nur für Tastenarten ! @ # \$ % ^ & * () _ - + = < [] { } , . / ? \ ' " ` ~ 0 bis 9 a bis z mit Unterscheidung Gross oder Klein Enter Leertaste und dient dem Erkennen von Tastaturkombinationen, die nicht vom Ereignis keydown abgefangen werden der Aufruf des keypress-Handlers ist nur möglich, wenn der keydown-Handler return true; bzw. beim IE alternativ event.returnValue=true; liefert
onkeyup	Tag <A>, , <TEXTAREA> Ereignis ausgelöst, wenn gedrückte Taste jeder Art losgelassen wird wird automatisch nach Ereignis keypress ausgelöst Ereignis keypress automatisch nach Ereignis keydown ausgelöst besitzt vordefiniertes Schlüsselwort KEYUP für captureEvents also document.captureEvent(Event.KEYUP); Tag <A>, , <TEXTAREA>

4.3.2.2.5.5.8.2.2. Tastatur-Eventeigenschaften

4.3.2.2.5.5.8.2.2.1. Steuertasten Alt, Strg (Ctrl) und Umschalt (Shift) (.modifiers)



.modifiers Bitmaske für Zustand der Steuertasten Shift, Alt und Strg
 dient zur Ermittlung der Steuertaste per vordefinierter Maske
 durch bitweise UND-Verknüpfung mit der Bitmaske
 ergibt die Verknüpfung 1, also true, so Steuertaste gedrückt worden

vordefinierte Maske für

```
Alt-Taste "Event.ALT_MASK"
Strg-Taste "Event.CTRL_MASK"
Shift-Taste "Event.SHIFT_MASK"
Bsp: return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0);
// nicht logisches UND && sondern bitweises UND &
```

4.3.2.2.5.5.8.2.2.2. Eventwert-Eigenschaft (.which)

.which enthält ASCII-Code der Taste
 Umwandlung in String per String.fromCharCode(TastenKodeASCII)

4.3.2.2.5.5.8.2.2.3. Eventart-Eigenschaft (.type)

.type enthält die Event-Art z.B. abort

4.3.2.2.5.5.8.2.2.4. Eventquelle-Eigenschaft (.target)

.target Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde
 entweder im HTML-Tag definieren per onXXX = ""
 oder Bezug auf den logischen Namen des HTML-Elementes
 if (logischer_name = Ereignis.target)

4.3.2.2.5.5.8.2.3. Tastatur-Ereignis-Folge onkeydown und onkeypress

Standard: erst onkeydown
 dann onkeypress

modifiziert: wenn Eventhandler für keydown return false; liefert, so wird der Eventhandler für
 keypress nicht aktiviert

4.3.2.2.5.5.8.2.3.1. Tastatur-Ereignis onkeydown

onkeydown ausgelöst bei allen Tasten
 liefert den ASCII-Code in die Event-Eigenschaft
 .which ASCII-Code der gedrückten Taste
 wenn 0 so keine Taste gedrückt
 wobei bei Buchstaben unterschieden wird
 Umwandlung in String per String.fromCharCode(TastenKodeASCII)

Zusatzanalyse möglich per Event-Eigenschaft

.modifiers Bitmaske für Zustand der Steuertasten Shift, Alt und Strg
 dient zur Ermittlung der Steuertaste per vordefinierter Maske
 durch bitweise UND-Verknüpfung mit der Bitmaske
 ergibt die Verknüpfung 1, also true, so Steuertaste gedrückt worden

vordefinierte Maske für

```
Alt-Taste "Event.ALT_MASK"
Strg-Taste "Event.CTRL_MASK"
Shift-Taste "Event.SHIFT_MASK"
```

Bsp: return ((Ereignis.modifiers & Event.SHIFT_MASK) != 0);
 // nicht logisches UND && sondern bitweises UND &

Unterbindung des Aufrufes des keypress-Handlers:
 keydown-Handler muss return false; bzw. event.returnValue=false; liefern

4.3.2.2.5.5.8.2.3.2. Tastatur-Ereignis onkeypress

wird unmittelbar nach dem keydown-Ereignis ausgelöst und dient dem Erkennen von Tastaturkombinationen,
 die nicht vom Ereignis keydown abgefangen werden

der Aufruf des keypress-Handlers ist **nur** möglich,
 wenn der keydown-Handler return true; bzw. event.returnValue=true; liefert

wird ausgelöst **nur** bei den Tasten

```
! @ # $ % ^ & * ( ) _ - + = < [ ] { } , . / ? \ | ' " ` ~
0 bis 9
a bis z mit Unterscheidung Gross oder Klein
Enter
Leertaste
```

alle anderen Tasten mit dem Ereignis keydown abfangen

4.3.2.2.5.5.8.3. Beispiel zur Tastatur-Eventbehandlung beim IE und Netscape

```
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
```

```
var ns = document.layers ? true : false;
```



```

var ie = document.all ? true : false;

// ***** Funktionen für Tasten aller Art holen
function NS_IE_TastencodeHolen_ASCII(Ereignis)
{
    if (ie)
    {return event.keyCode;}
    else
    {
        if (ns)
        {return Ereignis.which;}
    }
}
function NS_IE_TastencodeHolen_String(TastencodeASCII)
{
    if (TastencodeASCII == 0)
    {return "";}
    else
    {return String.fromCharCode(TastencodeASCII);}
}

function IE_TasteDauerdruck_StatusHolen()
{return event.repeat;} // true, so Taste im Dauerdruck, sonst false

// ***** Funktionen für Shift-Taste
// liefern true für Shift-Taste gedrückt; sonst false

function NS_IE_ShiftTaste_StatusHolen(Ereignis)
{
    if (ie)
    {return event.shiftKey;}
    else
    {
        if (ns)
        {return (Ereignis.modifiers & Event.SHIFT_MASK)
            != 0;
        }
    }
}

function IE_ShiftLeftTaste_StatusHolen()
{return event.shiftLeft;}

function IE_ShiftRightTaste_StatusHolen()
{return (event.shiftKey & !event.shiftLeft);}

// ***** Funktionen für Alt-Taste analog zu Shift aber für NS mit Event.ALT_MASK

// ***** Funktionen für Strg-Taste analog zu Shift aber für NS mit Event.CTRL_MASK

// -->
</SCRIPT>

```

4.3.2.2.5.5.9. Mouse-Eventbehandlung des IE und NS

Die Programmierung der Tastatur-Events unterscheidet sich nicht nur zwischen den Browsern IE und Netscape, sondern auch noch innerhalb deren Versionsnummern. Dieser Umstand zeugt vom divergierenden Ansatz der Browserhersteller, welcher im krassen Widerspruch zu einer Programmierungsfreundlichkeit steht.

Die bessere Konformität zur konsequent-objektorientierten Umsetzung des HTML-Dokumentes, seiner Elemente und Ereignisse hat der Internet Explorer, der damit ein weiteres Spektrum an Möglichkeiten zur Programmierung unter Javascript bietet, obwohl Microsoft weiterhin Visual Basic-Script als Komponente des Windows Script Host (Betriebssystem-Komponente) favorisiert und eine Dokumentierung der Javascript-Komponente (inklusive Debugger-Freeware-Tool zum aktuellen IE) konsequent vernachlässigt.

4.3.2.2.5.5.9.1. Mouse-Eventbehandlung beim Internet Explorer ab 4.x

4.3.2.2.5.5.9.1.1. Mouse-Eventarten

onclick	Ereignis ausgelöst, wenn Maus das Objekt angeklickt hat: Taste drücken und loslassen
	im Handler return false; kodieren für Links, Formularelemente und DIV wenn eine Aktion nicht erwünscht ist Bsp.: bei Link wird dieser nicht ausgeführt, obwohl angeklickt
	Tag <A>, <BODY>, <INPUT>, <DIV>
oncontextmenu	Ereignis ausgelöst direkt vor Aufruf des Kontextmenüs mit der rechten Maustaste



		Unterdrückung, wenn der Eventhandler event.returnValue=false; enthält (mit return false; erfolgt keine Unterdrückung) gilt standardgemäß HTML-seitenweit, also für alle Elemente der Seite, da das Ereignis standardgemäß bis nach oben weitergereicht wird
ondbclick	Tag	fast alle Ereignis ausgelöst, wenn mit linker Maustaste ein Doppelklick auf Objekt erfolgt 1 Klick = Maustaste drücken und loslassen
onmousedown	Tag	fast alle Ereignis ausgelöst mit Drücken irgendeiner Maustaste siehe Event-Eigenschaft .button return false; bzw. returnValue=false; unterbinden nicht die die Ausführung des Eventhandlers also Unterbindung z.B. in click oder contextmenu kodieren bei verschachtelt Objekten wird immer vom innen nach außen (unten nach oben) das Ereignis geliefert, solange bis das Event behandelt wurde kann per event.cancelBubble das Weiterreichen nach oben verhindert werden (in allen Ebene, die oberhalb derjenigen Ebene liegt, in der cancelBubble aktiviert wurde)
	Tag	<A>, <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.
onmouseenter		Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt Tag unbekannt wahrscheinlich ab IE 5.02 Empfehlung: mouseover verwenden
onmouseleave		Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt Tag unbekannt wahrscheinlich ab IE 5.02 Empfehlung: mouseout verwenden
onmousemove		Ereignis bei jeder Mausbewegung ausgelöst Auswertung für den Bereich eines Objektes durch dessen Eventhandler Tag <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.
onmouseout		Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt Tag <A>, <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.
onmouseover		Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt Tag <A>, <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.
onmouseup		Ereignis ausgelöst mit Loslassen irgendeiner Maustaste siehe Event-Eigenschaft .button return false; bzw. returnValue=false; unterbinden nicht die die Ausführung des Eventhandlers also Unterbindung z.B. in click oder contextmenu kodieren bei verschachtelt Objekten wird immer vom innen nach außen (unten nach oben) das Ereignis geliefert, solange bis das Event behandelt wurde kann per event.cancelBubble das Weiterreichen nach oben verhindert werden (in allen Ebene, die oberhalb derjenigen Ebene liegt, in der cancelBubble aktiviert wurde)
	Tag	<A>, <APPLET>, <AREA>, <BODY>, <DIV>, <FORM>, , <INPUT>, <MAP>, <SELECT>, , <TEXTAREA>, die meisten Textelemente wie etc.
4.3.2.2.5.5.9.1.2. Mouse-Event-Eigenschaften		
.button	0	keine Maustaste gedrückt
	1	linke Maustaste gedrückt
	2	rechte Maustaste gedrückt
	4	mittlere Maustaste gedrückt
	Kombination aus 1 bis 4 für Maustastenkombination	
	z.B. 3 = linke UND rechte Maustaste gedrückt (1 + 2 = 3) 7 = alle Maustasten gedrückt (1 + 2 + 3 + 4 = 7)	
.clientX	horizontale Pixel-Position der Maus gegen über linke obere Ecke (0,0) des Anzeigebereiches im Browserfenster	
aus	Hinweis: horizontale Pixel-Position der Maus gegenüber der linken oberen Ecke des HTML-Dokumentes ergibt sich aus event.clientX + document.body.scrollLeft	
.clientY	vertikale Pixel-Position der Maus gegen über linke obere Ecke (0,0) des Anzeigebereiches im Browserfenster	
	Hinweis: vertikale Pixel-Position der Maus gegenüber der linken oberen Ecke des HTML-Dokumentes ergibt sich aus event.clientY + document.body.scrollTop	



.fromElement	enthält Zeiger desjenigen Objektes, das das Ereignis onmouseout hat siehe auch .toElement
.offsetX	horizontale Pixel-Position der Maus im HTML-Element z.B. <DIV>, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0)
.offsetY	vertikale Pixel-Position der Maus im HTML-Element z.B. <DIV>, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0)
.returnValue	enthält den Rückkercode des Eventhandlers Eventhändler muss true liefern (return true;), um eine Aktion aufgrund des Events zuzulassen false liefern (return false;), um eine Aktion aufgrund des Events nicht zuzulassen Anwendung z.B. bei RESET und SUBMIT vom Formular
.screenX	horizontale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms
.screenY	vertikale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms
.srcElement	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per onXXX = "....." oder Bezug auf den logischen Namen des HTML-Elementes if (logischer_name = event.srcElement)
.toElement	enthält Zeiger desjenigen Objektes, das Ereignis onmouseover hat siehe auch .fromElement
.type	enthält die Event-Art z.B. abort
.x	horizontale Pixel-Position der Maus bezüglich der linken oberen Ecke (0,0) des Eltern-HTML-Elementes (parent)
.y	vertikale Pixel-Position der Maus bezüglich der linken oberen Ecke (0,0) des Eltern-HTML-Elementes (parent)

4.3.2.2.5.9.2. **Mouse-Eventbehandlung beim Netscape ab 4.x**

Der Event-Handler muss als Parameter das gewünschte Event übergeben bekommen

4.3.2.2.5.9.2.1. **Mouse-Eventarten**

onclick	Ereignis ausgelöst, wenn Maus das Objekt angeklickt hat: Taste drücken und loslassen im Handler return false; kodieren für Links, Formularelemente wenn eine Aktion nicht erwünscht ist Bsp.: bei Link wird dieser nicht ausgeführt, obwohl angeklickt
ondblclick	Tag <A>, <BODY>, <INPUT> Ereignis ausgelöst, wenn durch linke bzw. rechte Maustaste ein Doppelklick auf Objekt erfolgt 1 Klick = Maustaste drücken und loslassen
onlosecapture	Tag fast alle Ereignis ausgelöst, wenn releaseCapture() für mousemove, mouseover und mouseout ausgeführt wurde der User mit der Maus das Browserfenster verlässt
onmousedown	Tag <A>, <APPLET>, <BODY>, <DIV>, <FORM>, <INPUT>, <P>, <SELECT>, , <TABLE>, <TD>, <TEXTAREA>, <TR> Ereignis ausgelöst mit Drücken der linken oder rechten Maustaste (mittlere nicht !) siehe Event-Eigenschaft .which auch in Verbindung mit Umschalt (Shift) und Alt return false; unterbindet immer die die Ausführung des Eventhandlers return false bei rechter Maustase auf Object document wird das Kontextmenü gesperrt
onmousemove	Tag <A>, <DIV>, <INPUT TYPE="button"> Ereignis bei jeder Mausbewegung ausgelöst Auswertung für den Bereich eines Objektes durch dessen Eventhandler
onmouseout	Tag bei NS unbekannt Ereignis ausgelöst, wenn Maus den Bereich des Objektes verlässt
onmouseover	Tag <A>, <DIV>, Ereignis ausgelöst, wenn Maus den Bereich des Objektes betritt
onmouseup	Tag <A>, <DIV>, Ereignis ausgelöst mit Loslassen der linken Maustaste (rechte und mittlere nicht !) siehe Event-Eigenschaft .which return false; unterbindet immer die die Ausführung des Eventhandlers
	Tag <A>, <DIV>, <INPUT TYPE="button">
4.3.2.2.5.9.2.2. Mouse-Event-Eigenschaften	
.height	Höhe in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde
.layerX	horizontale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis darf kein resize-Event sein
.layerY	neue Breite in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Breite verändert wurde, nur resize-Event vertikale Pixel-Position der Maus im Layer, in dem das Ereignis ausgelöst wurde, bezüglich dessen linker oberer Ecke (0,0) Ereignis darf kein resize-Event sein
.pageX	neue Höhe in Pixel desjenigen Fensters in Pixel, das durch Mausziehen in der Höhe verändert wurde, nur resize-Event
.pageY	horizontale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
.screenX	vertikale Position der Maus bezüglich der linken oberen Ecke (0,0) des HTML-Dokumentes
	horizontale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms



.screenY	vertikale Pixel-Position der Maus gegenüber linker, oberer Ecke (0,0) des Bildschirms		
.target	Zeiger auf dasjenige HTML-Element, für das das Ereignis ausgelöst wurde entweder im HTML-Tag definieren per onXXX = "" oder Bezug auf den logischen Namen des HTML-Elementes if (logischer_name = event.srcElement)		
.type	enthält die Event-Art z.B. abort		
.which	enthält bei gedrückter	Maustaste:	1 für linke Taste 2 für mittlere Taste 3 für rechte Taste
.width	Breite in Pixel vom Fenster bzw. Frame, in dem ein Event ausgelöst wurde		
.x	identisch mit layerX		
.y	identisch mit layerY		

4.3.2.2.5.5.10. Eventbehandlung für Textoperationen mit der Windows-Zwischenablage (Clipboard)

nur Internet Explorer ab 5.x

4.3.2.2.5.5.10.1. Eventarten

onbeforecopy	markierten Text in die Zwischenablage kopieren: Ereignis, das direkt vor der Kopieraktion ausgelöst wird Tag alle Tags mit markierbarem Text z.B. <A>, <DIV>, , <TEXTAREA>		
onbeforecut	markierten Text ausschneiden und in die Zwischenablage einfügen: Ereignis, das direkt vor dem Ausschneiden ausgelöst wird verlangt im Händler event.returnValue = false; für Erzeugung der Kontextmenü-Eintrages zur Ausschneide-Operation (standardmäßig ist Ausschneiden nicht möglich) Tag alle Tags mit markierbarem Text z.B. <A>, <DIV>, , <TEXTAREA>		
onbeforepaste	Text aus Zwischenablage einfügen: Ereignis, das direkt vor dem Einfügen ausgelöst wird verlangt im Händler event.returnValue = false; für Erzeugung der Kontextmenü-Eintrages zur Einfüge-Operation (standardmäßig ist Einfügen nicht möglich) Tag alle Tags mit markierbarem Text z.B. <A>, <DIV>, , <TEXTAREA>		
oncopy	Einfüge-Operation im Kontextmenü ermöglichen: Handler muss return false; bzw. returnValue=false; liefern markierten Text in die Zwischenablage kopieren: Ereignis, das direkt nach der Kopieraktion ausgelöst wird Tag alle die Text definieren, <A>		
oncut	markierten Text ausschneiden und in die Zwischenablage einfügen: Ereignis, das direkt mit dem Ausschneiden ausgelöst wird verlangt im Handler event.returnValue = false; für Aktivierung des Kontextmenü-Eintrages zum Ausschneiden (standardmäßig ist Ausschneiden nicht möglich) Tag alle die Text definieren, <A>		
onpaste	Text aus Zwischenablage einfügen: Ereignis, das direkt mit dem Einfügen ausgelöst wird verlangt im Handler event.returnValue = false; für Aktivierung des Kontextmenü-Eintrages zum Einfügen (standardmäßig ist Einfügen nicht möglich) Tag alle Tags mit markierbarem Text z.B. <A>, <DIV>, , <TEXTAREA>		

4.3.2.2.5.5.10.2. Beispiel

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
var Text = "";

function AusschneidenUndInDieZwischenAblageEinfuegen()
{
    event.returnValue=false; // Kontextmenü-Eintrag aktivieren
    Text=Span_ID_Quellobjekt.innerText;
    Span_ID_Quellobjekt.innerText = ".... ups, der alte Text ist tatsaechlich weg ! ....";
}

function AusDerZwischenAblageEinfuegenInFormVonAnhaengen()
{
    event.returnValue = false; // Kontextmenü-Eintrag aktivieren
    Span_ID_Zielobjekt.innerText = Span_ID_Zielobjekt.innerText + Text + "<BR>";
}

```




```
function VorDemAusschneidenKontextMenuEintragErzeugen()
{event.returnValue=false;}

function VorDemEinfuegenKontextMenuEintragErzeugen()
{event.returnValue=false;}
//-->
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="Span_ID_Quellobjekt" onbeforecut="VorDemAusschneidenKontextMenuEintragErzeugen()"
oncut="AusschneidenUndInDieZwischenAblageEinfuegen()"
>
Diesen Text mit linker Maus markieren und mit Kontext-Menue (rechte Maus) ausschneiden !
</SPAN>
<BR>
<BR>
<BR>
<SPAN ID="Span_ID_Zielobjekt" onbeforepaste="VorDemEinfuegenKontextMenuEintragErzeugen()"
onpaste="AusDerZwischenAblageEinfuegenInFormVonAnhaengen()"
>
Bitte die Maus auf diesen Text setzen und Einfügen mit Kontext-Menue (rechte Maus) ausführen !<BR>
Es wird damit eine 3. Zeile erzeugt !<BR>
</SPAN>
</BODY>
</HTML>
```

4.3.2.2.5.5.11. Druck-Eventbehandlung nur Internet Explorer ab 5.x

onbeforeprint Druck eines Fensters oder Frames: Ereignis, das direkt vor dem Druckstart ausgelöst wird

Tag <BODY>, <FRAMESET>

onafterprint Druck eines Fensters oder Frames: Ereignis, das direkt nach dem Druckende ausgelöst wird

Tag <BODY>, <FRAMESET>

4.3.2.2.5.5.12. HTML-Element-Lade-Eventbehandlung des IE und NS

4.3.2.2.5.5.12.1. Lade-Ereignisse des Internet Explorer ab 4.x bzw. 5.x

4.3.2.2.5.5.12.1.1. Lade-Ereignisse für Bild

onabort Ladevorgang eines Bildes bricht ab z.B. wegen Useraktion Stop

Tag

onload Ereignis ausgelöst, wenn Bild vollständig geladen wurde zu animiertes Bild (Gif-Datei):

Ereignis wird bei jeder Animationswiederholung ausgelöst es kann nur pro komplette Animation das Ereignis behandelt werden

Tag <A>, <APPLET>, <BODY>, <DIV>, <FRAMESET>, <IFRAME>, , <SCRIPT>

4.3.2.2.5.5.12.1.2. Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)

onbeforeunload Verlassen der HTML-Seite: Ereignis, das direkt vor dem Verlassen ausgelöst wird wenn im Handler event.returnValue = 'freier_melde_text'; so wird automatisch vor dem Verlassen ein Anfragefenster geöffnet z.B. Anfrage, ob die HTML-Seite wirklich verlassen werden soll

Tag <BODY>, <FRAMESET>

onload Ereignis ausgelöst, wenn vollständig geladen wurde HTML-Dokument mit all seinen Elementen jeder Art Framset (innerhalb <FRAMESET> kodieren) DIV Applet, Script, Link, IFRAME

Tag <A>, <APPLET>, <BODY>, <DIV>, <FRAMESET>, <IFRAME>, , <SCRIPT>

onunload Ereignis ausgelöst, wenn das HTML-Dokument verlassen wird durch Schliessen Browserfenster Wechsel zu anderer Seite auch per Browser-Button window.document.open() window.document.write()

Tag <BODY>, <FRAMESET>

onstop Ereignis ausgelöst, wenn Stop-Button des Browsers gedrückt wurde Seite verlassen wird auch durch Browser-Buttons nur ab 5.x

4.3.2.2.5.5.12.2. Lade-Ereignisse des Netscape ab 3.x

4.3.2.2.5.5.12.2.1. Lade-Ereignisse für Bild



onabort	Ladevorgang eines Bildes bricht ab z.B. wegen Useraktion Stop
Tag	
onload	Ereignis ausgelöst, wenn Bild vollständig geladen wurde zu animiertes Bild (Gif-Datei): Ereignis load bei jedem Bildwechsel der Animation ausgelöst es kann also für jedes Bild das Ereignis behandelt werden
Tag	<BODY>, <DIV>, <FRAMESET>,

4.3.2.2.5.12.1.2. Lade-Ereignisse für HTML-Dokument und dessen Elemente (außer Bild)

onload	Ereignis ausgelöst, wenn vollständig geladen wurde HTML-Dokument mit all seinen Elementen jeder Art Framset (innerhalb <FRAMESET> kodieren) DIV Layer
Tag	<BODY>, <DIV>, <FRAMESET>,
onunload	Ereignis ausgelöst, wenn das HTML-Dokument verlassen wird durch Schliessen Browserfenster Wechsel zu anderer Seite auch per Browser-Button window.document.open() window.document.write()
Tag	<BODY>, <FRAMESET>

4.3.2.2.6. window.external Objekt des Internet Explorer

Dieses Objekt betrifft direkt die Privatsphäre des Users und ist mit Sorgfalt zu verwenden ! Jeder Interessenkonflikt zwischen dem Webseitenanbieter, der dieses Objekt nutzt, und dem User ist zu vermeiden. Die Nutzung des Objektes ist dem User transparent zu machen, so dass er die Wahl hat, ob das Objekt wirksam werden soll oder nicht.

Das Objekt ermöglicht den Zugriff auf

- das Kontextmenü
- die Favoritenliste
- den Microsoft Active Desktop (Desktop muss vom User per Desktop-Eigenschaften als Active Desktop eingestellt worden sein)
- die Microsoft Active Channel (Channel muss vom User per Desktop-Eigenschaften als Active Channel eingestellt worden sein)
- das Autocomplete bei Formularen (Autocomplete kann vom User in den Browser-Optionen eingerichtet werden)
- das permanente Speichern von User-Daten
- das Autoscans von Urls
- die Suche in Webseiten
- die Einstellungen zu Sprache, Favoriten, Sicherheit

kompletter Zugriff erst ab IE 5.x

Eigenschaften:

.menuArguments Zeiger auf dasjenige offene Fenster, indem ein Eintrag aus dem Kontextmenü ausgeführt wurde

Beispiel:

```
<SCRIPT LANGUAGE = "JavaScript">
var ZeigeraufWindow = window.external.menuArguments;

// beispielsweise einen selektierten Text im Fenster verarbeiten
var ZeigerAufDokuemntImWindow = ZeigeraufWindow.document;
var SelektionImDokument = ZeigerAufDokuemntImWindow.selection;
var TextBereichImDokument = SelektionImDokument.createRange();
var SelektierterTextImTextBereich = TextBereichImDokument.text;

if (SelektierterTextImTextBereich.length == 0)
{ TextBereichImDokument.text = "Einfuege Text"; }
else
{ TextBereichImDokument.text = SelektierterTextImTextBereich.toUpperCase(); }
</SCRIPT>
```

Methoden:

.AddChannel() Dialogbox zur Channel-Einstellung öffnen um einen Channel zu installieren (Microsoft Active Channel)
erzeugt im Fehlerfall eine Dialogbox und das Event onerror

Beispiel:

```
window.external.AddChannel("http://test/file.cdf");
```

.AddDesktop() eine Webseite oder Bild zum installierten Microsoft Active Desktop hinzufügen
wenn Active Desktop nicht installiert, so passiert nichts

Beispiel:

```
window.external.AddDesktopComponent("http://www.test.de","website",100,100,200,200);
```

.AddFavorite() Dialogbox zum Hinzufügen einer Url in die Favoritenliste für den User öffnen
Syntax:

```
logischer_window_name.external.AddFavorite(Kette1 [, Kette2])
```

Kette1

String

URL des Favoriteneintrages, also diejenige Url
die gebokkmarkt werden soll



Kette2

String

freier Vorgabe-Text des Favoriteneintrages
wird von Usereingabe in die Dialogbox
überschrieben

logischer_window_name

Zeiger laut open()

liefert nichts

Beispiel:

```
window.external.AddFavorite(location.href, document.title);
```

.AutoCompleteSaveForm()

permanentes Speichern von Daten eines Formulars in den AutoComplete-Data Store

Diese Methode ist mit Vorsicht zu genießen und richtet sich nach dem aktuellen Einstellungen von AUTOCOMplete, die der User in den Browser-Optionen treffen kann.

Gespeichert werden die Werte von INPUT's im Formular, die das Attribut NAME besitzen,
wobei auch INPUT TYPE=password speicherbar ist

Hinweis: Im Formular-Tag ist ebenfalls NAME zu kodieren

NAME-Attribut ist auch Voraussetzung für das Senden von Formulardaten an den
Server der Webseite

AutoComplete-Data Store kann vom User gelöscht werden (auch komponentenweise).

Beispiel:

```
<SCRIPT>
function Speichern()
{
    // erst speichern
    window.external.AutoCompleteSaveForm(ID_Formular);
    // dann löschen
    ID_Formular.ID_Input1.value="";
    ID_Formular.ID_Input2.value="";
}
</SCRIPT>
<FORM NAME="ID_Formular">
    Dieser Text wird gespeichert:
    <INPUT TYPE="text" NAME="ID_Input1">
    Dieser Text wird nicht gespeichert:
    <INPUT TYPE="text" NAME="ID_Input2" AUTOCOMplete="off">
</FORM>
<INPUT TYPE=button VALUE="Speichern" onclick="Speichern()">
```

.AutoScan()

Url einer beliebigen Webseite festlegen, die als Fehlermeldung vom Autoscan angezeigt wird

Webseite wird aufgerufen, wenn Autoscan nicht erfolgreich war
erklärt den Misserfolg

muss immer erreichbar, also anzeigbar sein

Standardwebseite auf Festplatte des User-PC vorhanden (falls der User diese
nicht manuell gelöscht hat)

Autoscan: Suchen nach einer anzuzeigenden Web-Seite im Internet durch den Browser
nach Eingabe der Url (auch mit Autovervollständigung)

Voraussetzungen für Autoscan:

es muss aktiv sein

Autoscan leider nur möglich für Domains mit

"www" am Anfang der Url

Suffixe der Url laut Registry-Eintrag

HKEY_LOCAL_MACHINE\software\microsoft\internet
explorer\main\urltemplate

(standardgemäß steht dort .com, .org, .net, und .edu)

Suche in der Reihenfolge der Einträge laut dem Registry-Eintrag

Hinweis: Der Registry-Eintrag ist manuell änderbar:

Bsp.: .de als Eintrag hinzufügen an gewünschte Position

Syntax:

logischer_window_name.external.AutoScan(Kette1, Kette2 [, Kette3])

Kette1

String

Url-Teil **nach** wwwund **vor** dem Suffix wie z.B.

.com, .org, .net, oder .edu

(siehe oben)

Kette2

String

Url der Webseite, die bei Fehler angezeigt werden
soll

Webseite kann im Internet liegen

muss immer erreichbar sein

Standard: IE-Fehlermeldung-Webseite lokal auf
der Festplatte des User-PC



	Kette3	String Referenz auf dasjenige Fenster, in dem die Webseite der Fehlermeldung angezeigt wird "_parent" "_self" (Standard) "_top" "_main"
	logischer_window_name	Zeiger laut open()
	liefert leider nichts	
Beispiel:	<pre>window.external.AutoScan("test", "error.htm", "_main");</pre> <p>für www.test.xxx</p>	mit xxx als Domain-Suffix laut Registry-Eintrag (siehe oben)
.ImportExportFavorites()	Import und Export der Favoritenliste aus dem/ in das Netscape-Bookmark-Format User muss Import/Export bestätigen per Dialogbox es wird HTTP benutzt Ziel-bzw. Quellort: Es wird durch Import immer hinzugefügt Es wird nach Export nicht gelöscht. Syntax: logischer_window_name.external.ImportExportFavorites(Wert, Kette)	
	Wert	true., so importieren false, so exportieren
	Kette	String Pfad des Ziel- (bei Export) bzw. Quellortes (bei Import) auf der User-PC-Festplatte bzw. im User-Netzwerk Server-Url auch möglich Url muss mit http:// beginnen Ort muss bereits vorhanden sein (kein automatisches Anlegen) wenn Leerkette, so wird automatisch eine Dialogbox für User-Auswahl geöffnet
	logischer_window_name	Zeiger laut open()
	liefert nichts	
Beispiel:	<pre>window.external.ImportExportFavorites(true, "http://www.test.com");</pre>	
.IsSubscribed()	auf Verfügbarkeit der Unterschrift zu einer Microsoft Active Channel-Datei prüfen (Channel Definition Format-Datei mit Dateisuffix ist *.cdf) nur für Dateien in gemeinsamer Domain, also nicht domain-übergreifend liefert immer Scriptfehler, wenn Domain-Wechsel durch Url der CDF-Datei erzeugt wird	
.NavigateAndFind()	Webseite laden, dann Text dort suchen und wenn gefunden, so diesen in der Webseite markieren (analog zu CTRL-F in der Webseite für Suchen und Finden) auch Suche in Frames bzw. Unterseiten der Webseite Syntax: logischer_window_name.external.NavigateAndFind(Kette1, Kette2, Kette3)	
	Kette1	String Url der Webseite oder lokaler Pfad also z.B. auch http:// oder c:\
	Kette2	String zu findender Text
	Kette3	String Referenz auf Frame/Fenster in der Webseite laut URL Leerkette möglich
	logischer_window_name	Zeiger laut open()
	liefert nichts	
Beispiel	<pre><HEAD> <SCRIPT> function Suchen() {</pre>	



```

        window.external.NavigateAndFind( "http://www.test.de/file.htm",
                                         ID_Select.options[ID_Select.selectedIndex].text,
                                         ""
                                         );
    }
</SCRIPT>
</HEAD>
<BODY>
    bitte selektieren
    <SELECT ID="ID_Select" onchange="Suchen()">
        <OPTION>Eintrag1
        <OPTION>Eintrag2
    </SELECT>
</BODY>

```

.ShowBrowserUI() öffnen einer IE-Dialogbox (UI = User-Interface) bezüglich
 Spracheinstellungen
 oder Favoritenverwaltung
 oder Sicherheitseinstellungen
 Inwieweit eine **geöffnete** Dialog-Box dann per Windows Script Host z.B. per VBScript programmierbar ist,
 wird hier nicht betrachtet.
 Ein User, dem diese Boxen durch eine Webseite geöffnet werden, sollte stets misstrauisch ein.

Beispiel:

```
<BUTTON onclick="window.external.ShowBrowserUI('LanguageDialog', null)">Spracheinstellungen</BUTTON>
```

4.3.2.2.7. window.history Objekt (history Collection)

Dieses Objekt ist eigentlich eine Collection.

Erzeugung: keine, da vom Browser erzeugt

Zugriff: **logischer_window_name.history.eigenschaft**
logischer_window_name.history.methode()

logischer_window_name.history[index].eigenschaft
logischer_window_name.history[index].methode()

index: ab 0
 Nummer des History-Eintrages im Dokument

Eigenschaften (Auswahl):

.current Zeichenkette mit Url der aktuellen Seite
 nur NS ab 3.x mit signiertem Script
 .length Anzahl der History-Einträge
 nur lesen
 .next Zeichenkette mit Url der nächsten Seite (falls vorhanden)
 nur NS ab 3.x mit signiertem Script
 .previous Zeichenkette mit Url der vorhergehenden Seite (falls vorhanden)
 nur NS ab 3.x mit signiertem Script

Methoden (Auswahl):

.back() vorhergehende Seite laut previous laden
 .forward() nächste Seite laut next laden
 .go(position) Position der zu ladenden Seite innerhalb der History
 0 entspricht erste geladene Seite
 history.length-1 entspricht aktuelle Seite
 Bsp.: onClick="window.history.go(0)"

history Object des Internet Explorer:

ist eigentlich eine Collection:

beinhaltet die Urls, die der User auf dem Client besucht hat (entspricht einem Verlauf des Surfens des Users)
 allerdings sortiert nach Urls und dort nach zugehörigen Unter-Urls (nicht sequentiell)

Urls können sequentiell-relativ zur aktuell besuchten Seite nur durch die Methoden des history-Objektes verwendet werden
 wobei Sprünge möglich sind

anhand eines Index 0 = aktuelle Seite
 < 0 zuvorliegende Seiten
 > 0 nachliegende Seiten

nur nutzbar wenn aktuelle Seite (Index 0) durch Rückwärtsgehen
 in der History eingestellt wurde

Alle Seiten werden immer aus dem Browser-Cache gelesen, wenn
 im HTML-Dokument nichts anderes vereinbart wurde
 in den Internet-Optionen des Browsers nichts anderes vereinbart wurde

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection
 ist der Index zum history-Objekt



Methoden:

.back()	Aktivierung einer zur aktuell besuchten Seite im Verlauf des Surfers davorliegenden Seite
.forward()	Aktivierung einer zur aktuell besuchten Seite im Verlauf des Surfers danachliegenden Seite
.go()	Aktivierung irgendeiner Url aus dem Verlauf
	entspricht beliebiger Selektion aus der Verlaufsliste

Beispiel: Reload des Dokumentes nach Resize des Browserfensters

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function neuLaden()
    {
        if (document.all)
        {history.go(0);}
    }
// -->
</SCRIPT>
</HEAD>
<BODY onResize=neuLaden();">
</BODY>
```

4.3.2.2.8. window.location Objekt

Erzeugung: keine, da vom Browser erzeugt

Zugriff: innerhalb eines Eventhandler:
 logischer_window_name.location.eigenschaft
 logischer_window_name.location.methode()
 sonst auch möglich
 location.eigenschaft
 location.methode()

Eigenschaften (Auswahl):

.hash	entspricht #hashtext zum Anspringen eines Ankers hier den Ankernamen mit vorgesetztem # ablegen
.host	entspricht hostname:port lesen und schreiben
.hostname	entspricht nur hostname lesen und schreiben
.href	gesamter Url (kompletter Url) zum Anspringen eines Ankers hier den Ankernamen ohne vorgesetztem # ablegen
.pathname	entspricht aus url /dateiname bzw. /dateiname#hashtext bzw. /dateiname?searchtext
.port	lesen und schreiben entspricht port
.protocol	lesen und schreiben entspricht Protokoll mit Doppelpunkt z.B. "http:" lesen und schreiben
.search	entspricht ?searchtext lesen und schreiben

Methoden (Auswahl):

.assing(url)	HTML-Dokument laden, nur IE ab 4.x
.reload([true])	wenn true entfällt: Dokument vom Cache auf Festplatte laden wenn true kodiert: Dokument vom Server und nicht Cache laden Achtung: Server kann selbst Cache haben und daraus das Dokument liefern
.replace(url)	aktuelle Seite mit neuer geladener Seite überschreiben sowie den HistoryEintrag zur aktuellen Seite überschreiben (keinen neuen bilden) --> BACK-Button wechselt danach nicht zur überschriebenen Seite, da diese in der History nicht mehr bekannt ist

location Objekt des Internet Explorer:

Container der Informationen der aktuellen Url

Eigenschaft .href enthält die komplette Url-Information und ist die Standard-Eigenschaft (entspricht HREF-Attribut):

location='http://www.test.de' ist gleichwertig mit location.href='http://www.test.de'

Bsp.: Quelltext anzeigen: Öffnet lokalen Standard-Text-Editor z.B. Notepad

<INPUT TYPE="button" VALUE="Quelltext ansehen" onclick="window.location = 'view-source:' + window.location.href">

Eigenschaften:

.hash	Teil des Wertes der Eigenschaft .href also Teil der Url als Anker
-------	---



	Teil folgt direkt hinter dem Nummernkreuz # und ohne Nummernkreuz
.host	Hostname und Port einer Location oder Url in der Form " hostname:port "
.hostname	Hostname einer Location oder Url in der Form " hostname:port " kann Domain oder IP sein
.href	Ziel-Url oder Anker als Sprungziel (z.B. <A>-Tag) siehe auch Eigenschaften .rel und .rev
.pathname	Datei und Pfad eines Objektes
.port	Port einer Location oder Url in der Form " hostname:port " basierend auf dem aktuellen Protokoll laut Eigenschaft .protocol z.B. <u>Standard-Ports</u> <u>Protokoll</u> 21 FTP 80 HTTP
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern
.search	Teil des Wertes des Eigenschaft .href Teil folgt direkt dem Fragezeichen wird als Querystring oder Formdata bezeichnet hat nichts mit der Suche auf Webseiten zu tun Hinweis: Fragezeichen-Anhang an der HREF dient zur Übergabe von String-Werten einer Webseite an eine andere: Quellseite besitzt HREF mit " ...?....." ruft Zielseite mit diesem HREF auf Zielseite wurde von Quellseite aufgerufen liest den Teil von HREF hinter dem ? als Zeichenkettendaten
Methoden:	
.assign()	neues Dokument zuweisen und laden im Verlauf (History) wird ein neuer Eintrag hinzugefügt (history Objekt) Altes Dokument ist per Vorwärts- und Zurück-Button einstellbar.
.reload()	aktuelles Dokument neu laden
.replace()	neues Dokument zuweisen und laden im Verlauf (History) wird der Eintrag des alten, vorherigen Dokumentes durch den Eintrag des neuen zu ladenden Dokumentes ersetzt. Altes Dokument ist damit nicht mehr per Vorwärts- und Zurück-Button einstellbar.

4.3.2.2.9. window.navigator Objekt

Container der Informationen über den Browser, Betriebssystem, regionale Einstellung des Users, CPU und Java-Maschine.

Die Informationen sind z.T. browserherstellerspezifisch und können sich von Version zu Version ändern. Es ist daher zu empfehlen, bei einer Browserprüfung auch Javascript zu verwenden und browsersepezifische Funktionen aufzurufen. Deren Rückkehrcode liefert den Beweis, ob der gewünschte Browser auch benutzt wird vom User (siehe DOM). Ein typisches Tarnverhalten zeigt der Opera-Browser: Er gibt sich als Internet Explorer aus.

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

navigator.eigenschaft
navigator.methode()

4.3.2.2.9.1. navigator Objekt im Netscape

4.3.2.2.9.1.1. Eigenschaften

alle Eigenschaften sind nur lesbar

.appName	Browser-Codename z.B. "Mozilla"
.appName	Browsersname z.B. "Netscape"
.appVersion	Plattform und Browserversion Aufbau: versionsnummer (system; land) Bsp: "3.0B2 (Win16;I)" "4.1 (WinNT;I)" Haupt-Versionsnummer ermittelbar per parseFloat(navigator.appVersion) z.B. 4 des IE 4.1 Minor-Version --> siehe .appMinorVersion
.cookieEnabled	wenn true so Cookiesverwaltung aktiv wenn false so Cookies abgeschaltet
.language	Browser-Sprachversion z.B. "en" "de"
.mimeTypes	Zeiger auf Feld aller im Browser verfügbaren Mimetypen Index kann Typ als Zeichenkette sein z.B. "image/png"
.oscpu	nur NS 6.x in der Dokumentation nicht weiter behandelt
.platform	Browser-Betriebssystem z.B. "Win32" oder "Win16"
.plugins	Zeiger auf Feld aller im Browser verfügbaren Plugins
.product	nur NS 6.x in der Dokumentation nicht weiter behandelt
.productSub	nur NS 6.x in der Dokumentation nicht weiter behandelt



.securityPolicy	Sicherheitseinstellungen vom Netscape nur NS 6.x in der Dokumentation nicht weiter behandelt
.userAgent	Browser-Codename UND -Version Bsp: "Mozilla / 3.0B2 (Win16;I)"
.vendor	nur NS 6.x in der Dokumentation nicht weiter behandelt
.vendorSub	nur NS 6.x in der Dokumentation nicht weiter behandelt

4.3.2.2.9.1.2. Methoden

.javaEnabled()	liefert true, wenn Browser Java kann und Java nicht abgeschaltet ist
.preference()	Benutzereinstellungen des Browsers verändern benötigt Privileg-Manger
.savePreferences()	aktuelle Benutzereinstellungen des Browsers sichern benötigt Privileg-Manger

4.3.2.2.9.1.3. navigator.plugins Collection des Netscape

Feld der Zeiger aller Plugins

Feldeintrag ist true, wenn Plugin im Feld vorhanden ist, also der Browser das Plugin besitzt.

Erzeugung:

keine, da vom Browser bereitgestellt

Zugriff:

navigator.plugins[index].eigenschaft
navigator.plugins[index].methode()

index: Zeichenkette z.B. "Shockwave"
Integer Index ab 0
muss in [] kodiert sein

Beispiel: Auf Adobe Acrobat-Reader-Plugin prüfen

```
<BODY>
  <SCRIPT LANGUAGE="JavaScript1.2">
    <!--
      if (navigator.plugins["Adobe Acrobat"] != null)
      {
        document.write("<EMBED SRC='test.pdf' WIDTH=600 HEIGHT=800>");
        document.write("<NOEMBED> .....<NOEMBED>");
      }
      else
      {
        document.write("Kein Adobe-Plugin !"); }
    //-->
  </SCRIPT>
</BODY>
```

Beispiel: Alle Plugins auflisten

```
<BODY>
  <SCRIPT LANGUAGE="JavaScript1.2">
    <!--
      for (i=0; i < navigator.plugins.length; i++)
      {
        document.writeln(navigator.plugins[i].name);
        document.writeln(navigator.plugins[i].description);
        document.writeln(navigator.plugins[i].filename);
      }
    //-->
  </BODY>
```

Eigenschaften:

.description	Zeichenkette der Plugin-Kurzbeschreibung als String
.filename	Plugin-Dateiname als String
.length	Anzahl der Plugins, ab 1
.mimeTypes	Zeiger auf navigator.mimeType Collection Beispiel für Zeigerbezug: navigator.plugins.mimeType[index].eigenschaft mit index String Mimetype Integer als Index ab 0 muss in [] kodiert sein
.name	Name des Plugin als String

Methoden:

.refresh()	Funktionswert true HTML-Dokument mit seinem per EMBED eingebetteten Objekten neu laden falls Plugin noch nicht installiert ist, so es dabei installieren false kein Refresh
------------	--



.taintEnabled() Data-Tainting (Daten verwerfen) per navigator Objekt

4.3.2.2.9.1.4. navigator.mimeTypes Collection

Feld aller Mimetypen also Dateitypen, die Browser für Plugin und Dateiverarbeitung verarbeiten kann bzw. soll. Als Index dient der Mimetyp als String oder ein Integerwert ab 0.

Ein Plugin wird einem Mimetyp zugeordnet.

Hinweis: Ab dem IE 6.0 werden keine Plugins mehr unterstützt. Es muss ein Active-X-Control verwendet werden

Die Eigenschaft .enablePlugin enthält den Zeiger auf das installierte Plugin (sonst null-Zeiger). Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.

Das Speichern der Zeiger **aller** Plugin-Objekte erfolgt in der Collection navigator.plugins.

Feldelement:

Reihenfolge der Feldelemente ist browserspezifisch.

Es besteht die Möglichkeit, dass Mimetypen, die keine Plugins repräsentieren, ebenfalls mit in der Collection liegen

Bsp.: "image/jpeg".

Beispiel für einen Mimetyp für echten Plugin: "application/pdf" für Adobe Acrobat-Plugin

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

navigator.mimeTypes[mime_typ].eigenschaft

mime_typ: String, der als Feldindex dient
z.B. "text/html"
muss in [] kodiert sein

Beispiel 1: Feld auslesen

```
for (var Index = 0; Index < navigator.mimeTypes.length; Index ++)  
{ alert( navigator.mimeTypes[Index].type + "\n"  
+ navigator.mimeTypes[Index].suffixes + "\n"  
+ navigator.mimeTypes[Index].description  
);  
}
```

Beispiel 2: Beschreibung zum Plugin anzeigen

```
if (navigator.mimeTypes["application/pdf"])  
{alert(navigator.mimeTypes["application/pdf"].description);}
```

Beispiel 3: Daten an Plugin für VRML-Dateiverarbeitung übergeben

```
if (navigator.mimeTypes["x-world/x-vrml"])  
{  
    if(navigator.mimeTypes["x-world/x-vrml"].enablePlugin != null)  
    {  
        document.write('<OBJECT DATA="zyeplin.wrl" WIDTH="400" HEIGHT="300"></OBJECT>');  
    }  
}
```

Beispiel 4: Suffix prüfen

```
if (navigator.mimeTypes["image/jpeg"])  
{alert(navigator.mimeTypes["application/pdf"].suffixes);}
```

Eigenschaften:

.description	Zeichenkette mit der Kurzbeschreibung des Mimetyps
.enablePlugin	Zeiger auf ein Plugin null-Zeiger, wenn Plugin nicht installiert ist Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.
.length	Anzahl der Feldelemente, ab 1
.suffixes	Liste aller erlaubten Dateisuffixe zum Mimetyp
.type	mime_typ z.B. "text/html"

Methoden:

keine

4.3.2.2.9.2. navigator Objekt im Internet Explorer

4.3.2.2.9.2.1. Eigenschaften

.appName	Codename des Browsers per navigator Objekt
.appMinorVersion	Update der Browserversion per navigator Objekt
.appName	Name des Browsers per navigator Objekt
.appVersion	Betriebssystem-Plattform und Browserversion per navigator Objekt
.browserLanguage	Sprache des Betriebssystems und nicht des Browsers (Browsersprache kann andere sein als die des Betriebssystems z.B. französischer Browser auf deutschem Windows)



.cookieEnabled	Cookiesstatus lesen aber nicht verändern per navigator Objekt
.cpuClass	CPU-Klasse per navigator Objekt
.mimeTypes	Zeiger auf Collection mimeTypees
.onLine	Onlinestatus des Browsers ermitteln per navigator Objekt
.platform	keine Überprüfung auf Netzwerkverbindung
.plugins	Name des Betriebssystems per navigator Objekt
.systemLanguage	Zeiger auf Collection plugin
.userAgent	Standard-Sprache des Betriebssystems (Sprache der Installation) per navigator Objekt
.userLanguage	Browser-Codename und -Version per navigator Objekt
.userProfile	vom User eingestellte Sprache des Betriebssystems (nicht die Sprache der Installation, sondern die regionale Sprache des Betriebssystems)
	Zeiger auf Collection userProfile

4.3.2.2.9.2.2. Methoden

.javaEnabled()	prüfen auf generelle Ausführbarkeit von Javacode, also ob Java-Maschine im Browser aktiv ist per navigator Objekt
.taintEnabled()	Data-Tainting (Daten verwerfen) per navigator Objekt

4.3.2.2.9.2.3. navigator.mimeType Collection (auch bei IE 6.x)

Feld aller Mimetypen also Dateitypen, die Browser für Plugin und Dateiverarbeitung verarbeiten kann bzw. soll. Als Index dient der Mimetyp als String oder ein Integerwert ab 0.

Ein Plugin wird einem Mimetyp zugeordnet.

Ab dem IE 6.0 werden keine Plugins mehr unterstützt. Es muss ein Active-X-Control verwendet werden.

Die Eigenschaft .enablePlugin enthält den Zeiger auf das installierte Plugin (sonst null-Zeiger). Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.

Das Speichern des Zeiger **aller** Plugin-Objekte erfolgt in der Collection navigator.plugins.

Feldelement:

Reihenfolge der Feldelemente ist browserspezifisch.

Es besteht die Möglichkeit, dass Mimetypen, die keine Plugins repräsentieren, ebenfalls mit in der Collection liegen

Bsp.: "image/jpeg".

Beispiel für einen Mimetyp für echten Plugin: "application/pdf" für Adobe Acrobat-Plugin

Beim Internet Explorer muss diese Collection nicht komplett gefüllt sein, kann aber (probieren!).

Ab dem IE 6.0 werden keine Plugins mehr unterstützt. Es muss ein Active-X-Control verwendet werden.

Es ist empfehlenswert, beim Internet Explorer **nur** mit der document.embeds Collection zu arbeiten. Die Collection navigator.mimeType kann zwar unter NS und IE mit dem gleichen Script-Code angesprochen werden, aber das ist spätestens ab IE 6.0 nicht mehr möglich.

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

navigator.mimeType[mime_typ].eigenschaft

mime_typ: String, der als Feldindex dient
z.B. "text/html"

Beispiele 1: Feld auslesen

```
for (var Index = 0; Index < navigator.mimeType.length; Index++)
{ alert( navigator.mimeType[Index].type + "\n"
+ navigator.mimeType[Index].suffixes + "\n"
+ navigator.mimeType[Index].description
);
}
```

Beispiel 2: Beschreibung zum Plugin anzeigen

```
if (navigator.mimeType["application/pdf"])
{alert(navigator.mimeType["application/pdf"].description);}
```

Beispiel 3: Daten an Plugin für VRML-Dateiverarbeitung übergeben

```
if (navigator.mimeType["x-world/x-vrml"])
{
    if(navigator.mimeType["x-world/x-vrml"].enablePlugin != null)
    {
        document.write('<OBJECT DATA="zyeplin.wrl" WIDTH="400" HEIGHT="300"></OBJECT>');
    }
}
```

Beispiel 4: Suffix prüfen

```
if (navigator.mimeType["image/jpeg"])
```



```
{alert(navigator.mimeTypes["application/pdf"].suffixes);}
```

Eigenschaften:

<code>.description</code>	Zeichenkette mit der Kurzbeschreibung des Mimetyps
<code>.enablePlugin</code>	Zeiger auf ein Plugin
<code>.length</code>	null-Zeiger, wenn Plugin nicht installiert ist
<code>.suffixes</code>	Mit diesem Zeiger sind die Eigenschaften und Methoden des Plugins ansprechbar. Jedes Plugin hat eigene Eigenschaften und Methoden.
<code>.type</code>	Anzahl der Feldelemente, ab 1
<code>.mime_type</code>	Liste aller erlaubten Dateisuffixe zum Mimetyp
<code>.mime_type</code>	mime_type z.B. "text/html"

Methoden:

keine

4.3.2.9.2.4. navigator.plugins Collection des Internet Explorer (auch bei IE 6.x)

Diese Collection dient nur für Kompatibilität mit anderen plugin-fähigen Browsern und stellt ein Alias der document.embeds Collection dar.

Die Collection `document.embeds` ist ein Alias für die `plugins` Collection nur bezüglich von Plugins (und nicht anderen einbettbaren Objekten), wobei letztere nur der Kompatibilität mit anderen plugin-fähigen Browsern dient.

Das Ansprechen von Plugins kann über die Collection navigator.mimeTypes per Eigenschaft .enabledPlugin erfolgen, die einen Zeiger auf das installierte Plugin enthält (wenn nicht installiert so null.Zeiger). Diese Collection muss aber beim Internet Explorer nicht komplett gefüllt sein, kann aber (ausprobieren). Wenn der Zeiger nicht null ist, dann sind die plugin-eigenen Methoden und Eigenschaften über Punktnotation ansprechbar. Beim Internet Explorer muss diese Collection nicht komplett gefüllt sein , kann aber (probieren !).

Es ist empfehlenswert, beim Internet Explorer **nur** mit der document.embeds Collection zu arbeiten. Die Collection navigator.mimeTypes kann zwar unter NS und IE mit dem gleichen Script-Code angesprochen werden, aber das ist spätestens ab IE 6.0 nicht mehr möglich.

Achtung: Ab IE 6.x werden keine Plugins mehr unterstützt, damit ist die plugins Collection hinfällig. Inwieweit diese Collectionen noch implementiert ist oder bleibt, ist durch den Programmierer zu prüfen. Ab dem IE 6.x muss jedes Plugin, das in das Dokument eingebunden werden soll, durch ein ActiveX-Control implementiert werden. Plugins, wie sie beim Netscape existieren, laufen unter dem IE ab 6.x nicht mehr.

Syntax:

[var ZeigerAufFeld =] navigator.plugins	
[var ZeigerAufFeldElement =] navigator.plugins[Index]	
Index	Integer ab 0
oder	String Name oder ID des Elementes
	(analog zu NAME und ID-Attribut)
	Name des Plugins
	muss in [] kodiert sein
ZeigerAufFeldElement	
	ist null, wenn Feldelement nicht vorhanden

Beispiel: Auf Adobe Acrobat-Reader-Plugin prüfen

```
<BODY>
  <SCRIPT LANGUAGE="JavaScript1.2">
    <!--
      if (navigator.plugins["Adobe Acrobat"] != null)
      {
        document.write("<EMBED SRC='test.pdf' WIDTH=600 HEIGHT=800>");
        document.write("<NOEMBED> .....</NOEMBED>");
      }
      else
      {
        document.write("Kein Adobe-Plugin !"); }
    //-->
  </SCRIPT>
</BODY>
```

Eigenschaften:

<u>Eigenschaften:</u>	
.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

<code>.item()</code>	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <code><INPUT TYPE=image ...></code> da dafür die children-Collection verwendet werden muss !!!
<code>.namedItem()</code>	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
<code>.tags()</code>	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

4.3.2.2.9.2.5. navigator.userProfile Objekt des Internet Explorer

Objekt zur nur lesenden Verwaltung von User-Profile-Informationen (vCard-Daten)

ab IE 4.x

Die vCard-Daten werden beim Internet Explorer im Objekt document.form verwertet, wenn im Formular das AutoComplete aktiviert ist.



"vCard.Business.City Business"	Stadt für vCard.Business.City-Schema
"vCard.Business.Country Business"	Land/Region für vCard.Business.Country-Schema
"vCard.Business.Fax Business"	Faxnummer für vCard.Business.Fax-Schema
"vCard.Business.Phone Business"	Telefonnummer für vCard.Business.Phone-Schema
"vCard.Business.State Business"	Staat, Provinz, Gebiet für vCard.Business.State-Schema
"vCard.Business.StreetAddress Business"	Strassenname für vCard.Business.StreetAddress-Schema
"vCard.Business.URL Business"	Webadresse für vCard.Business.URL-Schema
"vCard.Business.Zipcode Business"	Postleitzahl für vCard.Business.Zipcode-Schema
"vCard.Cellular"	Einzelanschluss-Telefonnummer für vCard.Cellular-Schema
"vCard.Company"	Firmenname (Firma) für vCard.Company-Schema
"vCard.Department"	Unternehmensname oder -teilname für vCard.Department-Schema
"vCard.DisplayName"	nutzerspezifischer angezeigter Name für vCard.DisplayName-Schema
"vCard.Email"	E-mail Adresse für vCard.Email-Schema
"vCard.FirstName"	Vorname für vCard.FirstName-Schema
"vCard.Gender"	Geschlecht für vCard.Gender-Schema
"vCard.Home.City"	Name der Heimatstadt für vCard.Home.City-Schema
"vCard.Home.Country"	Heimatland/-region für vCard.Home.Country-Schema
"vCard.Home.Fax"	Faxnummer zu Hause für vCard.Home.FAX-Schema
"vCard.Home.Phone"	Telefonnummer zu Hause für vCard.Home.Phone-Schema
"vCard.Home.State"	Heimatstaat/Provinz/Gebiet für vCard.Home.State-Schema
"vCard.Home.StreetAddress"	Heimat-Strassenname für vCard.Home.StreetAddress-Schema
"vCard.Home.Zipcode"	Heimat-Postleitzahl für vCard.Home.Zipcode-Schema
"vCard.Homepage"	private Webadresse für vCard.Homepage-Schema
"vCard.JobTitle"	Berufsbezeichnung für vCard.JobTitle-Schema
"vCard.LastName"	Nachname für vCard.LastName-Schema
"vCard.MiddleName"	zweiter Vorname für vCard.MiddleName-Schema
"vCard.Notes"	Bemerkungen für vCard.Notes-Schema
"vCard.Office"	Büroort für vCard.Office-Schema
"vCard.Pager"	Cityrufnummer für vCard.Pager-Schema
"vCard.GenderXXX"	mit XXX laut oben z.B. Notes

Hinweis: AutoComplete betrifft auch Werte laut VALUE-Attribut von Textfeldern im Formular, aber NUR, wenn diese Felder auch das NAME-Attribut besitzen. VALUE-Werte werden automatisch für die automatische Wiederverwendung im Browser gespeichert (auch bei Password-Feldern !!!). Daher sollte die Nutzung von AutoComplete reiflich überlegt sein.
Das nachträgliche Löschen von per AutoComplete automatisch gespeicherte Werte ist in den Internet-Optionen des IE vollziehbar.

Beispiel 1:

```
// Request Queue füllen

// es soll der Wert zu "vcard.displayname" ermittelt werden
navigator.userProfile.addReadRequest("vcard.displayname");
// es soll der Wert zu "vcard.gender" ermittelt werden
navigator.userProfile.addReadRequest("vcard.gender");

// Datenermitteln per Queue
navigator.userProfile.doReadRequest(12, "Daten von unbekannt verwendet");

// Queue leeren
navigator.userProfile.clearRequest();
```

Beispiel 2:

```
// lesen vom Wert zu "vcard.displayname"
var Name = navigator.userProfile.getAttribute("vcard.displayname");

// lesen vom Wert zu "vcard.gender"
var Gender = navigator.userProfile.getAttribute("vcard.gender");
```

Eigenschaften:

keine

Methoden:

.addReadRequest()	Eintrag in Queue für Lesezugriff (Request Queue) erzeugen
.clearRequest()	Queue für Lesezugriff (Request Queue) leeren
.doReadRequest()	Lesezugriff auf alle vCard-Datenarten, die laut aktueller Request Queue vorgegeben sind
.getAttribute()	Lesezugriff auf einzelnen vCard-Wert per .getAttribute()
.setAttribute()	einzelnen vCard-Wert lesen ohne Request Queue
	Wert von vorhandenem Attribut setzen
	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert

4.3.2.2.10. window.popup Objekt des Internet Explorer

Objekt für Popup-Fenster z.B. für Dialog, Meldungen, Tooltip
ab IE 5.5

Diese Fensterform ist eine stark reduzierte Instanz des window Objektes, wobei der Inhalt per Script kodiert werden muss (z.B. HTML-Tags).



Ein PopUp-Fenster

wird aktuell, wenn es angezeigt wird

wird automatisch geschlossen, sobald ein anderes Objekt den Fokus erhält, also aktiv wird

z.B. durch User-Klick außerhalb des PopUp-Fensters

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende



Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popublocker hat ein Popufenster geblockt. Sie können den Popublocker deaktivieren oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popublocker einschalten
weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popublocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popublocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popublocker verwaltet wird.

Der Popublocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popublocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popublockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus
onblur
onfocusin
onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popublockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

window.focus();

window.document.focus();

if(document.body!=null)

{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)

{document.body.focus();}

}

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

popupzeiger.show(...);

Hinweis: Der Popufehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus

.setActive() ist Teilmenge von .focus(): nur das aktiv setzen

funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLETT, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.

Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.

Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.



Ein Control, das programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp (http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte



SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.



Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen:• 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popuptmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen.



Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87='parent.Y_unload(0)'; X86[0]=new Function(",X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet,
so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr
wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
    unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild

<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12) ">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12) ">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,
innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

Erzeugung:

.createPopup() Popupfenster ohne irgendwelche Fensterelemente öffnen z.B. für Anzeige von Tooltips
Fenster wird mit der Anzeige per Methode .show() zum aktuellen Fenster
erst geschlossen, wenn **anderes Fenster** aktuell wird
z.B. durch Klicken außerhalb des Popupfensters

ab IE 5.5

Syntax:

```
[ var Zeiger = ] logischer_window_name.createPopup()
```

Zeiger

Referenz auf das Popupfenster (Zeiger entspricht ID),
wird für die Verwendung der Eigenschaften und
Methoden benutzt per
Zeiger.eigenschaft
Zeiger.methode()

logischer_window_name

Zeiger laut open()

Zugriff:

```
zeiger_auf_popup_fenster.eigenschaft
zeiger_auf_popup_fenster.methode
```

zeiger_auf_popup_fenster

laut Erzeugung

Beispiel 1:

```
<SCRIPT LANGUAGE="JScript">
// erzeugen
var PopupFenster = window.createPopup(); // windows referenziert das aktuelle und instanzierte Fenster

// Körper erzeugen also body Objekt
var PopupFensterKoerper = PopupFenster.document.body;
```



```
// Körper füllen
PopupFensterKoerper.innerHTML = "Das ist ein Popup-Fenster";

// alles anzeigen
PopupFenster.show(100, 100, 200, 50, document.body);
</SCRIPT>
```

Beispiel 2 für diverse Meldungsfenster per **jeweiligem** PopUp-Fenster
(es kann immer nur genau 1 Popup-Fenster angezeigt werden):

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">

    var PopUpFensterFeld = new Array();

    // nachfolgende Felder beschreiben die PopUp-Fenster und müssen
    // identische Anzahl der Feldelemente haben
    // Feld-Index ist die Nummer des PopUp-Fensters

    var PopUpFenster_RahmenStyle_Feld = new Array
    (
        "solid black 4px",
        "solid blue 3px",
        "solid green 2px"
    );

    var PopUpFenster_HintergrundFarbe_Feld = new Array
    (
        "yellow",
        "gray",
        "orange"
    );

    var PopUpFenster_Inhalt_Feld = new Array
    (
        "<B>Inhalt PopUpFenster 0<B>",
        "Inhalt PopUpFenster 1",
        "<TT>Inhalt PopUpFenster 2<TT>"
    );

    var PopUpFenster_X_Koordinate_Feld = new Array
    (
        100,
        150,
        200
    );

    var PopUpFenster_Y_Koordinate_Feld = new Array
    (
        150,
        200,
        250
    );

    var PopUpFenster_Breite_Koordinate_Feld = new Array
    (
        80,
        140,
        200
    );

    var PopUpFenster_Hoehe_Koordinate_Feld = new Array
    (
        100,
        140,
        180
    );

    var AnzahlPopUpFenster = PopUpFenster_Hoehe_Koordinate_Feld.length;

    function PopupFensterInstanzieren(NummerDesPopUpFensters, ZeigerAufElternFenster)
        // NummerDesPopUpFensters ab 0
```



```

    {
        PopUpFensterFeld[NummerDesPopUpFensters] =
            ZeigerAufElternFenster.createPopup();
    }

function PopupFensterFuellen(NummerDesPopUpFensters)
{
    // Body des Popup-Fensters gestalten
    var PopupFenster_Body =
        PopUpFensterFeld[NummerDesPopUpFensters].document.body;

    PopupFenster_Body.style.backgroundColor =
        PopUpFenster_HintergrundFarbe_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.style.border =
        PopUpFenster_RahmenStyle_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.innerHTML =
        PopUpFenster_Inhalt_Feld[NummerDesPopUpFensters];
}

function PopupFensterAnzeigen(NummerDesPopUpFensters,
    ObjektZuDemPopUpFensterRelativPositioniertIst
)
{
    // Popup-Fenster anzeigen und damit öffnen
    PopUpFensterFeld[NummerDesPopUpFensters].show(
        PopUpFenster_X_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Y_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Breite_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Hoehe_Koordinate_Feld[NummerDesPopUpFensters],
        ObjektZuDemPopUpFensterRelativPositioniertIst
    );
}

function PopupFensterSchliessen(NummerDesPopUpFensters)
{
    var Zeiger = PopUpFensterFeld[NummerDesPopUpFensters];

    if (Zeiger.isOpen)
    { Zeiger.hide(); }
}

function Init()
{
    // PopupFenster instanzieren im aktuellen Fenster (window)
    for (var i = 0 ; i < AnzahlPopUpFenster; i++)
    {
        PopupFensterInstanzieren(i, window);
        PopupFensterFuellen(i);
    }
}
</SCRIPT>
</HEAD>
<BODY onload="Init();">
    Durch Klick ausserhalb des Popup-Fensters wird dieses auch automatisch geschlossen.
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 anzeigen"
        onclick=" PopupFensterAnzeigen(0, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 anzeigen"
        onclick=" PopupFensterAnzeigen(1, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 anzeigen"
        onclick=" PopupFensterAnzeigen(2, document.body);"
    >
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 schliessen"
        onclick=" PopupFensterSchliessen(0);"

```



```

>
<INPUT TYPE=button
VALUE="PopUpFenster 1 schliessen"
onclick=" PopUpFensterSchliessen(1);"
>
<INPUT TYPE=button
VALUE="PopUpFenster 2 schliessen"
onclick=" PopUpFensterSchliessen(2);"
>
</BODY>
</HTML>

```

Hinweis: Es ist aufgrund mangelnder Eigenschaften nicht möglich, Daten von einem Popup-Fenster zu empfangen, die per Eventhandler übergeben werden sollen, der im PopUpfenster aufgerufen wird, aber im Dokument kodiert ist, das das PopUpfenster erzeugt. Leider existiert keine Eigenschaft `.opener`. Als Ersatz dient folgendes Beispiel, das ein normales Fenster benutzt:

```

function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ ' onclick="opener.OnClickHandler();" '
+ '>'
+ '</BODY>'
+ '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();

```

Eigenschaften:

<code>.document</code>	Zeiger auf das HTML-Dokument im Popup-Fenster, das per <code>.show()</code> instanziiert wird Es können damit alle Eigenschaften und Methoden des Objektes <code>document</code> referenziert werden siehe Objekt <code>window.popup</code> Syntax: <pre>[var Zeiger =] zeiger_auf_popup_fenster.document</pre>		
<code>.isOpen</code>	nur lesen prüfen ob ein per <code>.createPopup()</code> instanziiertes Popup-Fenster angezeigt wird siehe Objekt <code>window.popup</code> Syntax: <pre>[var Wert =] zeiger_auf_popup_fenster.isOpen</pre> <table border="0" style="margin-left: 100px;"> <tr> <td style="text-align: right;">Wert</td> <td>true, so angezeigt false, so nicht angezeigt</td> </tr> </table> <pre>zeiger_auf_popup_fenster</pre> laut Erzeugung per <code>.createPopup()</code>	Wert	true, so angezeigt false, so nicht angezeigt
Wert	true, so angezeigt false, so nicht angezeigt		

Methoden:

<code>.hide()</code>	nur lesen ein angezeigtes Popup-Fenster schliessen Hinweis: Das Popup-Fenster wird automatisch geschlossen, wenn ein anderes Objekt den Focus erhält bzw. aktiv wird, z.B. durch Klick des Users außerhalb des PopUpfensters. ändert Wert der Eigenschaft <code>.isOpen</code> siehe Objekt <code>window.popup</code> Syntax: <pre>zeiger_auf_popup_fenster.hide()</pre>
	<pre>zeiger_auf_popup_fenster</pre> laut Erzeugung per <code>.createPopup()</code> liefert nichts



<code>.show()</code>	ein per <code>.createPopup()</code> instanziiertes Popup-Fenster anzeigen
Hinweis:	Es kann immer nur genau 1 Popup-Fenster angezeigt werden.
	Das Popup-Fenster wird automatisch geschlossen, wenn ein anderes Objekt den Focus erhält bzw. aktiv wird, z.B. durch Klick des Users außerhalb des Popuppfensters
	ändert Wert der Eigenschaft <code>.isOpen</code>
	siehe Objekt <code>window.popup</code>
Syntax:	<code>zeiger_auf_popup_fenster. .show(Wert, Wert2, Wert3, Wert4 [, Zeiger])</code>
Wert1	X-Koordinate der linken oberen Fenstecke bezüglich eines Objektes oder des Bildschirms Koordinaten-Ursprung (0,0) des Bildschirms liegt in der linken oberen Ecke Integer, in Pixel, ≥ 0
Wert2	Y-Koordinate der linken oberen Fenstecke bezüglich eines Objektes oder des Bildschirms Koordinaten-Ursprung (0,0) des Bildschirms liegt in der linken oberen Ecke Integer, in Pixel, ≥ 0
Wert3	Breite des Fensters in Pixel Integer, > 0
Wert4	Höhe des Fensters in Pixel Integer, > 0
Zeiger	auf Objekt zu dem Wert1 und Wert2 relativ sind Standard: Bildschirm
<code>zeiger_auf_popup_fenster</code>	laut Erzeugung per <code>.createPopup()</code>
	liefert nichts

4.3.2.2.11. window.screen Objekt

Dieses Objekt beschreibt den Bildschirm

Die linke obere Browserfensterecke ist der Ursprung des Grafiksystems, also (0,0) mit (x,y)

x für horizontal

y für vertikal

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

`screen.eigenschaft`

4.3.2.2.11.1. screen Objekt im Netscape

Eigenschaften:

<code>.availHeight</code>	maximal verfügbare Fensterhöhe in Pixel laut max. Bildschirmauflösung
<code>.availLeft</code>	minimale Pixel-X-Position der Browserfensterecke links oben (horizontal) hängt von Position der Windows-Taskleiste ab
<code>.availTop</code>	minimale Pixel-Y-Position des Browserfensterecke links oben (vertikal) hängt von Position der Windows-Taskleiste ab
<code>.availWidth</code>	maximale verfügbare Fensterbreite in Pixel laut max. Bildschirmauflösung
<code>.colorDepth</code>	Farbtiefe der aktuellen Farbpalette, also Anzahl der verfügbaren Farben
<code>.height</code>	aktuelle Höhe in Pixel der Bildschirmauflösung
<code>.left</code>	Abstand links in Pixel
<code>.pixelDepth</code>	aktuelle Anzahl Farbbits pro Bildpunkt laut aktueller Bildschirmauflösung 1 oder 4 oder 8 oder 15 oder 16 oder 24 oder 32
<code>.top</code>	Abstand oben in Pixel
<code>.updateInterval</code>	Verzögerung der Anzeigegeschwindigkeit in Millisekunden z.B. bei Animationen nur IE
<code>.width</code>	aktuelle Breite in Pixel der Bildschirmauflösung

Methoden:

keine

4.3.2.2.11.2. screen Objekt im Internet Explorer

Eigenschaften:

<code>.availHeight</code>	verfügbare Höhe des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar per screen Objekt
<code>.availWidth</code>	verfügbare Breite des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar per screen Objekt
<code>.bufferDepth</code>	Anzahl der Bits pro Pixel für eine Farbe UND Verwendung des off-screen bitmap buffer per screen Objekt
<code>.colorDepth</code>	Anzahl der Bits pro Pixel für eine Farbe UND Verwendung destination device or buffer



	wird durch den Wert der Eigenschaft .bufferDepth überschrieben, wenn .bufferDepth mit Wert > 0
	per screen Objekt
.deviceXDPI	Aktuelle Anzahl der horizontalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
	per screen Objekt
.deviceYDPI	Aktuelle Anzahl der vertikalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
	per screen Objekt
.fontSmoothingEnabled	Fontglättung auf Bildschirm
	per screen Objekt
.height	Auflösung des Bildschirms in Höhe, also Anzahl der vertikalen Pixel
	per screen Objekt
.logicalXDPI	Standard-Anzahl der horizontalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
	per screen Objekt
.logicalYDPI	Standard-Anzahl der vertikalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
	per screen Objekt
.updateInterval	Refresh-Intervall des Bildschirms: aus dem Puffer neu schreiben
	per screen Objekt
.width	Auflösung des Bildschirms in Breite, also Anzahl der horizontalen Pixel
	per screen Objekt

Methoden:

keine

4.3.2.2.12. Sonstige Objekte und Collectionen (Auswahl)**4.3.2.2.12.1. regexp Objekt als Instanz des Script-Objektes RegExp**

RegExp dient der Definition eines regulären Ausdrucks für die Zeichenkettensuche, wobei dabei automatisch wird z.T. auch von Methoden anderer Objekte z.B. String für Zeichenkettensuche benutzt.

regexp ist eine Instanz des Objektes RegExp
dient der detaillierten Festsetzung des Suchmusters.

Erzeugung:

```
var regexp_name= suchmuster;
oder var regexp_name=new RegExp(suchmuster);
```

suchmuster: regulärer Ausdruck vom Objekttyp RegExp
verwendet für Suche
syntaktischer Aufbau:
/suchmuster_elemente_mit_kommatrennung/options

/ / sind zu kodierende Zeichen anstelle " " markieren ein Suchmuster und keine Zeichenkette

suchmuster_elemente: verwendet für detaillierte Musterdefinition
stehen innerhalb von / / (NICHT " ")
Kommatrennung:
ein Element kann zum Beispiel sein:

\zeichen hebt die ursprüngliche Bedeutung eines Zeichens auf
Bsp: + ist Addition oder Verketten
\+ ist das Zeichen '+'

^zeichenfolge markiert eine feste Zeichenfolge am **Anfang** des Suchbegriffes
Bsp: ^Otto entspricht "Otto Waalkes"
entspricht NICHT "Heinrich Waalkes"

\$zeichenfolge markiert eine feste Zeichenfolge am **Ende** des Suchbegriffes
Bsp: ^Waalkes entspricht "Otto Waalkes"
entspricht NICHT "Otto der Komiker"

zeichen* zeichen kann mehrfach auftreten (beliebig viel)
auch Null-Mal
Bsp: t* entspricht "tto Waalkes"
entspricht "to Waalkes"
entspricht "o Waalkes"

zeichen+ zeichen kann mehrfach auftreten (beliebig viel)
aber mindestens einmal
Bsp: t* entspricht "tto Waalkes"
entspricht "to Waalkes"
entspricht NICHT "o Waalkes"

zeichen? zeichen kann höchstens einmal oder keinmal auftreten
Bsp: t* entspricht NICHT "tto Waalkes"
entspricht "to Waalkes"
entspricht NICHT "o Waalkes"

. Punkt im Suchmuster ersetzt genau ein beliebiges Zeichen an seiner Position

zeichenfolge_1|zeichenfolge_2
entweder eine von beiden Zeichenfolgen
oder beide zusammen



[zeichen_menge]	alternative Zeichen als Menge einschliessend Bsp: [Oo] "[Oo]tto Waalkes" für "Otto Waalkes" oder "otto Waalkes"
[^zeichen_menge]	alternative Zeichen als Menge ausschliessen Bsp: [^O] "[^O]tto Waalkes" für "tto Waalkes" oder "otto Waalkes"
\bzeichenfolge	Worttrenner z.B. Blank oder Tab werden beachtet Bsp: "Otto\bWaalkes" für "Otto Waalkes" nicht für "OttoWaalkes"
\Bzeichenfolge	Worttrenner z.B. Blank oder Tab dürfen nicht enthalten sein Bsp: "Otto\BWaalkes" für "OttoWaalkes" nicht für "Otto Waalkes"
\dzeichenfolge	oder [0] [9] oder [0-9] Ziffer 0 bis 9 dürfen enthalten sein Bsp: "Otto Waalkes der \d" für "Otto Waalkes der 1." nicht für "Otto Waalkes der V." Bsp: "Otto Waalkes der [1]" für "Otto Waalkes der 1." nicht für "Otto Waalkes der 2." Bsp: "Otto Waalkes der [0-9]" für "Otto Waalkes der 1." für "Otto Waalkes der 2." nicht für "Otto Waalkes der V."
\Dzeichenfolge	oder [^0] [^9] oder [^0-9] Ziffer 0 bis 9 dürfen nicht enthalten sein Bsp: "Otto Waalkes der \D" für "Otto Waalkes der Erste" nicht für "Otto Waalkes der 1." Bsp: "Otto Waalkes der [^1]" für "Otto Waalkes der 2." nicht für "Otto Waalkes der 1." Bsp: "Otto Waalkes der [^0-9]" für "Otto Waalkes der Erste" nicht für "Otto Waalkes der 1." nicht für "Otto Waalkes der 2."
\fzeichenfolge	Seitenvorschub muss vorhanden sein vor zeichenfolge, wenn sie gefunden werden soll
\mzeichenfolge	Mehrzeiligkeit zulässig, nur NS ab 6.x
\nzeichenfolge	Zeilenvorschub muss vorhanden sein vor zeichenfolge, wenn sie gefunden werden soll
\rzeichenfolge	Wagenrücklauf muss vorhanden sein vor zeichenfolge, wenn sie gefunden werden soll
\tzeichenfolge	Tabulator muss vorhanden sein vor zeichenfolge, wenn sie gefunden werden soll
\vzeichenfolge	vertikaler Tabulator muss vorhanden sein vor zeichenfolge, wenn sie gefunden werden soll
\wzeichenfolge	zeichenfolge muss Ziffern und oder Buchstaben enthalten, wenn sie gefunden werden soll
\Wzeichenfolge	zeichenfolge darf keine Ziffer und oder Buchstaben enthalten, wenn sie gefunden werden soll
[f n r t v]zeichenfolge	zeichenfolge muss mindestens einen der in [] gesetzten Steuerzeichen enthalten, wenn sie gefunden werden soll
[^f n r t v]zeichenfolge	zeichenfolge darf keinen der in [] gesetzten Steuerzeichen enthalten, wenn sie gefunden werden soll
[A-Za-z0-9]zeichenfolge	zeichenfolge muss mindestens einen der in [] gesetzten Zeichen enthalten, wenn sie gefunden werden soll
[^A-Za-z0-9]zeichenfolge	zeichenfolge darf keinen der in [] gesetzten Zeichen enthalten, wenn sie gefunden werden soll
(suchmuster_element)	Bsp: ([0-9])
\szeichenfolge	zeichenfolge muss mindestens ein beliebigen Trenner z.B. Blank, Tab, Seitenvorschub etc. enthalten, wenn sie gefunden werden soll
\Szeichenfolge	zeichenfolge darf keinen Trenner z.B. Blank, Tab, Seitenvorschub etc.



enthalten, wenn sie gefunden werden soll

optionen: sind wahlweise kodierbar

i	ignoriere Groß und Klein bei der Suche
oder g	mehrfaches Auftreten möglich
oder gi	entspricht i UND g
Standard ist	Unterscheidung Groß und Klein
sowie	Suche NUR bis zum ersten Auffinden

Zugriff:

```
regex_name.eigenschaft
regex_name.methode()
```

Eigenschaften (Auswahl):

.global	ist true wenn Option g kodiert
.ignoreCase	ist true wenn Option i kodiert
.lastIndex	Angabe derjenigen Position, die direkt hinter dem ZULETZT gefunden Teilstring liegt, der dem Suchmuster entspricht
	Verwendung z.B. für Weitersuchen
.multiline	nur ab NS 6.x
	st true wenn Option m kodiert
.source	enthält das gesamte Suchmuster aber ohne "/"
	nur lesbar

Methoden (Auswahl):

.compile(suchmuster,options)	suchmuster --> siehe oben optionen --> siehe oben automatisch ausgeführt					
.exec(suchmuster)	auch kodierbar als regex(suchmuster) also ohne .exec füllt Eigenschaften von RegExp und regex, wobei RegExp.input die zu durchsuchende Kette enthält liefert Array mit index ab 0: <table border="0"> <tr> <td>als Position im Suchbegriff an der Übereinstimmung besteht</td> </tr> <tr> <td>0 letzte gefundene Übereinstimmung</td> </tr> <tr> <td>ab 1 bis unbegrenzt: jeweils gefundene Übereinstimmung mit den Teil des Suchbegriffes innerhalb von ()</td> </tr> <tr> <td>Feldelement enthält gefundene Übereinstimmung</td> </tr> <tr> <td>ist null-Zeiger wenn Auswertung fehlerhaft war</td> </tr> </table>	als Position im Suchbegriff an der Übereinstimmung besteht	0 letzte gefundene Übereinstimmung	ab 1 bis unbegrenzt: jeweils gefundene Übereinstimmung mit den Teil des Suchbegriffes innerhalb von ()	Feldelement enthält gefundene Übereinstimmung	ist null-Zeiger wenn Auswertung fehlerhaft war
als Position im Suchbegriff an der Übereinstimmung besteht						
0 letzte gefundene Übereinstimmung						
ab 1 bis unbegrenzt: jeweils gefundene Übereinstimmung mit den Teil des Suchbegriffes innerhalb von ()						
Feldelement enthält gefundene Übereinstimmung						
ist null-Zeiger wenn Auswertung fehlerhaft war						

Beispiel:

```
function getInfo()
{
    re = /(w+)\s(d+)/;
    var m = re.exec();
    alert(m[1] + ", your age is " + m[2]);
}

<FORM>
    <INPUT TYPE="text" NAME="Alter" onChange="getInfo(this);">
</FORM>
```

.test(suchmuster)	Suche laut suchmuster liefert true, wenn Suche erfolgreich (sonst false)
-------------------	---

Beispiel:

```
function testinput(re, str)
{
    if (re.test(str))
    {midstring = " contains ";}
    else
    {midstring = " does not contain ";}

    document.write (str + midstring + re.source);
}
```

Beispiel: <SCRIPT>

```
<!--
    // Vorbereitung der Suche
    var zu_durchsuchende_kette="Otto Waalkes";
    // oder RegExp.input="Otto Waalkes"

    var such_muster=/(\wOtto)/g;
    // such_muster ist ein regulärer Ausdruck
    // (Objekt RegExp)
    // Suchmuster ist Otto, wobei Otto gefunden
    // werden muss für eine erfolgreiche Suche
    // anstelle von " ist / zu kodieren
    // alternativ nicht möglich
    // RegExp.input="Otto"
    // dann kein detailliertes Suchmuster
```



```

// Option g alternativ kodierbar per
// RegExp.multiline=true;

// Start der Suche nur durch eine Zeichenketten-Funktion, die RegExp beachtet,
// z.B. durch replace() als Methode des Objektes String

var ergebnis_kette=zu_durchsuchende_kette.
    replace(such_muster,"Heinrich");

// Start der Suche nach Otto
// wenn gefunden, so durch Heinrich ersetzen
// Ergebnis der Suche und des Ersetzens
// nach ergebnis_kette bringen
// oder alternativ z.B.
// var ergebnis_kette=RegExp.lastMatch
// --> Suchmuster hat ( ), also auch $1 bis $9
// verwendbar

//oder bei exec() bzw. regexp(such_muster)
var ergebnis_array=such_muster.exec(zu_durchsuchende_kette);

//-->
</SCRIPT>

```

.toSource() liefert String mit Quellcode

.toString() liefert String des Objekthinhaltes

Beispiel

```

myExp = new RegExp("a+b+c");
alert(myExp.toString()) displays "/a+b+c/"

```

4.3.2.2.12.2. **RegExp Objekt (nicht regexp Objekt)**

Dieses Objekt ist eine Komponente der Scriptsprache.

RegExp dient der Definition eines regulären Ausdruckes für die Zeichenkettensuche, wobei dabei automatisch wird z.T. auch von Methoden anderer Objekte z.B. String für Zeichenkettensuche benutzt.

regexp ist eine Instanz des Objektes RegExp
dient der detaillierten Festsetzung des Suchmusters.

Erzeugung:

keine, da vom Browser erzeugt

Zugriff:

RegExp.eigenschaft

Beispiel: <SCRIPT>

```

<!--
// Vorbereitung der Suche
var zu_durchsuchende_kette="Otto Waalkes";
// oder RegExp.input="Otto Waalkes"

var such_muster=/Otto/g; // ist Objekt vom Typ regexp und NICHT RegExp
// Suchmuster ist Otto
// anstelle von " ist / zu kodieren
// alternativ auch möglich
// RegExp.input="Otto"
// dann aber ohne detailliertes Suchmuster

RegExp.multiline=true; // mehrfaches Vorkommen beachten

// Start der Suche nur durch eine Zeichenketten-Funktion, die RegExp beachtet,
// durch replace() als Methode des Objektes String
// Hinweis: Die Methode search() des Objektes String wertet nicht RegExp-
// Vorgaben aus

var ergebnis_kette=zu_durchsuchende_kette. replace(such_muster,"Heinrich");

// Start der Suche nach Otto
// wenn gefunden, so durch Heinrich ersetzen
// Ergebnis der Suche und des Ersetzens
// nach ergebnis_kette bringen
// oder alternativ z.B.
// var ergebnis_kette=RegExp.lastMatch
// --> Suchmuster hatte keine ( ), also
// z.B. $1 nicht verwendbar

//-->
</SCRIPT>

```



Eigenschaften (Auswahl):

sämtliche Steuerzeichen .\$ sind im NS ab 6.x deprecated

IE ab Version 5.5 verwendbar

.input oder .\$_	Zeichenkette, die durchsucht werden soll
.multiline oder .\$*	Wertzuweisung per RegExp.input="zeichenkette"
	auf true setzen für Suche über mehrere Zeilen
	auf false setzen für Suche nur bis zum nächsten Zeilenumbruch
.lastMatch oder .\$&	enthält Zeichenfolge, die zuletzt gefunden wurde und dem Suchbegriff (Suchmuster) entspricht
	ab IE 5.5
.lastParent oder .\$+	enthält die Zeichenfolge, die zuletzt gefunden wurde und dem Suchbegriff innerhalb ()
	aus regexp (NICHT RegExp !!!) entspricht
.leftContext oder .\$`	nicht .\$'
	enthält Teilkette, der LINKS neben dem gefundenen Muster steht
.rightContext oder .\$'	nicht .\$`
	enthält Teilkette, der Rechts neben dem gefundenen Muster steht
.\$1 bis .\$9	die letzten 9 gefundenen Zeichenketten, die dem Muster innerhalb ()
	aus regexp (NICHT RegExp) entsprechen
.\$01 bis .\$09	die letzten 100 gefundenen Zeichenketten, die dem Muster innerhalb ()
	aus regexp (NICHT RegExp) entsprechen

Methoden:

keine

4.3.2.2. 12.3. script Objekt des Internet Explorer

Container für Scriptcode z.B. Javascript oder XML

Achtung: Jedes Script hinter dem Ende-Tag vom FRAMESET-Element wird ignoriert und nicht geparkt !!

Der erste Script-Block mit .language Eigenschaft legt die Sprache aller nachfolgenden fest, wenn dort kein .language codiert wurde.

Fehlt generell die .language Eigenschaft, dann so wird die microsoft-spezifisches Scriptmaschine zum Parsen verwendet.

Bps.: JScript ist Microsoft-Javascript-Standard
 Javascript ist allgemeiner der Javascript-Standard

Zugriff:**document.all["id_des_scripts"].eigenschaft****document.all["id_des_scripts"].eigenschaft**

id_des_scripts laut ID-Attribut

Beispiel 1: für XML-Script

```
<HEAD>
  <SCRIPT LANGUAGE="Javascript">
    var XMLScriptObjekt = document.all["ID_XML"].XMLDocument;
  </SCRIPT>

  <SCRIPT LANGUAGE="XML" ID="ID_XML">
    .....
  </SCRIPT>
</HEAD>
```

Beispiel 2: für Javascript

```
<HEAD>
  <SCRIPT LANGUAGE="Javascript">
    var JavaScriptObjekt = document.all["ID_JavaScript"]
  </SCRIPT>

  <SCRIPT LANGUAGE="Javascript" ID="ID_JavaScript">
    .....
  </SCRIPT>
```

Eigenschaften:

.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.charset	Zeichensatz zum Encoden eines Objektes im Dokument
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
	ohne Rahmen
	ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
	ohne Rahmen
	ohne Scrollbalken
.defer	Pars-Status des Scriptes per script Objekt



	download eines Scriptes kann beschleunigt werden, wenn es vom Parsen zurückgestellt wird
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.event	On-Eventbezeichner mit angefügten Klammernpaar Script soll das Event verwalten per script Objekt immer mit Eigenschaft .htmlFor kodieren
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.htmlFor	Referenz auf Objekt, das das Event laut Eigenschaft .event verwalten soll und dafür einen Scriptaufruf im Ereignishandler onxxx besitzt per script Objekt immer mit Eigenschaft .event kodieren
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind nicht beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.text	Text eines Objektes
.type	Mimetype des Scriptes per script Objekt Mimetype wird von der Scriptmaschine zum Parsen verwendet Achtung: Wert muss passend zum Wert der Eigenschaft .language sein !
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.XMLDocument	Referenz auf XML-Dokument (XML-DOM)



Methoden:

<code>.addBehavior()</code>	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
<code>.applyElement()</code>	ab IE 5.x bis unter IE 5.5 Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode <code>.createElement()</code> erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft <code>.innerHTML</code> gelöscht !
<code>.attachEvent()</code>	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode <code>.detachEvent()</code> Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
<code>.clearAttributes()</code>	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
<code>.cloneNode()</code>	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.contains()</code>	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
<code>.detachEvent()</code>	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_bezeichner</code> zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
<code>.dragDrop()</code>	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
<code>.fireEvent()</code>	ein Event auslösen
<code>.getAdjacentText()</code>	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
<code>.getAttribute()</code>	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
<code>.getAttributeNode()</code>	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
<code>.getElementsByTagName()</code>	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei <code>document.all</code> Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementsById()</code> Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code>
<code>.hasChildNodes()</code>	DOM nicht geändert prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
<code>.insertAdjacentElement()</code>	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
<code>.mergeAttributes()</code>	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !!



.normalize()	DOM wird geändert Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
.removeAttributeNode()	DOM wird geändert entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
.removeBehavior()	DOM wird geändert per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
.replaceAdjacentText()	DOM wird geändert Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
.setAttribute()	DOM wird nicht geändert Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
.setAttributeNode()	DOM wird nur bei Erzeugung geändert Attribut einem Knoten zuweisen und Referenz liefern
.swapNode()	DOM wird geändert Positionen von 2 Knoten im DOM tauschen (Zeigertausch) nur sichtbar wenn Endetag geparkt DOM wird geändert

4.3.2.2.12.4. *document.scripts Collection des Internet Explorer*

Feld der Zeiger aller script Objekte im Dokument

Elementefolge laut HTML-Koding, also auch in der Reihenfolge HEAD --> BODY

Syntax:

```
[ var ZeigerAufFeld = ] document.scripts
[ var ZeigerAufFeldElement = ] document.scripts[Index [, SubIndex] ]
```

Index	Integer ab 0 oder String Name oder ID des Elementes (analog zu NAME und ID-Attribut) muss in [] kodiert sein
SubIndex	optional nur kodieren wenn Index ein String ist Integer als Unterindex also Unterelement eines Elementes
ZeigerAufFeldElement	ist null, wenn Feldelement nicht vorhanden wenn gleichnamige Script-Objekte vorhanden, so wird ein Zeiger auf die Collection aus diesen Script-Objekten geliefert

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.tags()	Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.12.5. *var Objekt des Internet Explorer*

Basisobjekt zu einer Variablen

Erzeugung:

in Script mit der Anweisung var deklariert werden kann (aber nicht muss)

in HTML mit dem VAR-Tag deklariert wird

Beispiel: Das ist eine HTML-<VAR>Variable</VAR>die mit italic font gerendert wird

Zugriff:

in Script per Bezeichner der Variablen als Referenz
ID-Attribut

in HTML per ID-Attribut

Eigenschaften:

.accessKey Tastaturzugriff auf ein Objekt per Alt + Taste
bei Ausführung der Tastenkombination wird
das Sprungziel wird focussiert
die Sprungquelle defocussiert



	das Focus-Ereignis ausgelöst
	vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
.innerHTML	Zeiger aus ID bilden var Zeiger = eval(object.id); Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objekthinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem



	des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master nur genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert



	<p>Element kann selbst Kinder haben</p> <p>Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde</p> <p>Achtung: Wenn Element per Methode <code>.createElement()</code> erzeugt wurde, aber nicht im Dokumentenbaum eingebunden ist, so wird die Eigenschaft <code>.innerHTML</code> gelöscht !</p>
<code>.attachEvent()</code>	<p>Einschalten des Registrieren eines Events durch Eventhandler</p> <p>Hinweis: Abschalten mit Methode <code>.detachEvent()</code></p> <p>Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)</p>
<code>.blur()</code>	<p>Element den Focus wegnehmen und Event <code>onblur</code> auslösen</p> <p>Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !</p> <p>vor IE 5.0 <code>TABINDEX</code>-Attribut muss kodiert sein</p> <p>ab IE 5.0 <code>TABINDEX</code>-Attribut muss nicht kodiert sein</p>
<code>.clearAttributes()</code>	<p>alle HTML-Attribute eines Objektes entfernen</p> <p>außer ID, STYLE und per Script definierte Attribute</p> <p>Script-erzeugte Attribute nicht entfernbar</p> <p>DOM wird geändert</p>
<code>.click()</code>	<p>simuliert einen Klick auf das Element und löst <code>onclick</code>-Event aus</p> <p>manipuliert nicht den Focus</p>
<code>.cloneNode()</code>	<p>Objekt klonen und Referenz des erzeugten Klone liefern</p> <p>DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)</p>
<code>.componentFromPoint()</code>	<p>Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt</p> <p>auch für CSS-Layout</p> <p><code>onmouseover</code>-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code></p> <p>also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben</p> <p>Overbereich der Maus ist mehr als 1 Pixel gross</p> <p>beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.</p>
<code>.contains()</code>	<p>prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist</p> <p>DOM nicht geändert</p>
<code>.detachEvent()</code>	<p>Abschalten des Registrieren eines Events durch Eventhandler</p> <p>wobei Registrierung mit Methode <code>.attachEvent()</code> aktiviert wurde</p> <p>Abschalten = ordnet dem Window-Objekt das Event laut Parameter <code>event_bezeichner</code> zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)</p>
<code>.fireEvent()</code>	<p>ein Event auslösen</p>
<code>.focus()</code>	<p>Focus setzen und Focus-Event auslösen</p> <p>nur nach dem kompletten Laden des Dokumentes</p> <p>vor IE 5.x: Objekt muss <code>TABINDEX</code>-Attribut besitzen</p>
<code>.getAdjacentText()</code>	<p>Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann</p> <p>Text kann HTML-Tags enthalten, muss aber nicht</p> <p>DOM nicht geändert</p>
<code>.getAttribute()</code>	<p>Wert eines per HTML erzeugten Attributes liefern</p> <p>DOM nicht geändert</p>
<code>.getAttributeNode()</code>	<p>Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf <code>attribute.name</code> Eigenschaft.</p> <p>Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht</p> <p>Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt</p> <p>Wert des Attributes wird somit über die Referenz laut DOM erreichbar</p> <p>DOM nicht geändert</p>
<code>.getBoundingClientRect()</code>	<p>Referenz auf <code>TextRectangle</code>-Objekt im Element holen</p>
<code>.getClientRects()</code>	<p>Referenz auf Feld der Zeiger auf <code>TextRectangle</code>-Objekte im Fenster</p> <p>Feld mit Index als Integer ab 0</p> <p>pro Eintrag ein Rectangle</p>
<code>.getElementsByTagName()</code>	<p>Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.</p> <p>Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei <code>document.all</code> Collection)</p> <p>Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !</p> <p>Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode <code>getElementsById()</code></p> <p>Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode <code>getElementsByName()</code></p>
<code>.getExpression()</code>	<p>DOM nicht geändert</p> <p>Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern</p> <p>Style-Eigenschaft ist per Methoden <code>expression()</code> oder <code>setExpression()</code> zu definieren</p> <p>DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)</p>
<code>.hasChildNodes()</code>	<p>prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt</p> <p>DOM nicht geändert</p>



.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.insertBefore()	Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
.removeAttribute()	Hinweis: einschalten per Methode .setCapture() entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeChild()	Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern sichtbar erst mit parsen des Endetags DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,



onmouseover und onmouseout.
 ab IE 5.5
 Hinweis: ausschalten per Methode .releaseCapture()
 .setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form
 objekt.style.eigenschaft.
 dient
 Ausdruck nur als Script kodierbar
 DOM wird nicht geändert
 .swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
 nur sichtbar wenn Endetag geparkt
 DOM wird geändert

4.3.2.2.12.6. xml Objekt im Internet Explorer

Dieses Objekt definiert eine Daten-Insel als eigenständige Datenmenge im HTML-Dokument, die per XML kodiert wird.

Daten
 sind dabei nur vom String-Typ
 werden nicht gerendert (angezeigt)
 liegen direkt im HTML-Dokument

ab IE 5.x

XML-Daten werden nach dem XML-DOM verarbeitet, siehe Objekt XMLHttpRequest (ab IE 7) und Objekt userProfile.

Erzeugung:

durch Browser

Zugriff:

solange das HTML-Dokument als Container instanziiert ist

Beispiel:

```
<XML ID="ID_DatenInsel">
  <METADATA>
    <AUTHOR>Ich </AUTHOR>
    <GENERATOR> Notepad</GENERATOR>
    <PAGETYPE>Reference</PAGETYPE>
    <ABSTRACT>Dateninsel </ABSTRACT>
  </METADATA>
</XML>
```

```
alert("Daten komplett geparkt = " + (ID_DatenInsel.readyState == "complete"));
```

Eigenschaften:

.canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf
 .id Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)
 ID-Attribut in HTML: Wert ist String alphanumerisch
 muss mit Buchstaben beginnen
 Unterstrich _ verwendbar
 kann in " " bzw. ' ' kodiert werden, muss aber nicht
 Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).
 Zeiger aus ID bilden var Zeiger = eval(object.id);
 Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
 .isContentEditable Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
 Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
 false Content nicht editierbar
 true Content editierbar
 .isDisabled Interaktionsfähigkeit
 nur wenn sichtbar so User-Interaktion möglich
 false User kann mit Objekt interagieren
 true User kann mit Objekt nicht interagieren
 .isMultiLine Mehrzeiligkeit des Objekthinhaltes
 false Objekt hat genau 1 Zeile
 true Objekt hat mindestens 1 Zeile
 .parentElement Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
 .readyState aktueller Status des Objektes beim Füllen des Objektes mit Daten
 "uninitialized" Objekt ist nicht initialisiert
 "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung
 "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert
 "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten
 "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
 .recordset Zeiger des Standard-Record-Set in einem Datasource-Objekt (DSO)
 Methode .namedRecordset() liefert den Zeiger für Datenquellen-Objekt mit Namen
 also eines beliebige Mitgliedes im Datasource-Objekt (DSO)
 .scopeName Namensraum laut XMLNS-Attribut
 .src Url der Daten z.B. vom Image in normaler Auflösung
 .tagUrn Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
 .XMLDocument Referenz auf XML-Dokument (XML-DOM)



Methoden:

<code>.addBehavior()</code>	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
<code>.componentFromPoint()</code>	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode <code>.componentFromPoint()</code> also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Überbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
<code>.fireEvent()</code>	ein Event auslösen true Event erfolgreich ausgelöst false Event nicht ausgelöst
<code>.getAttributeNode()</code>	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
<code>.namedRecordset()</code>	Zeiger desjenigen Datenquellen-Record-Objektes mit Namen, das den Standard-Record-Set enthält Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an Eigenschaft <code>.recordset</code> liefert den Zeiger für den Standard-Record-Set in einem Datasource-Objekt (DSO)
<code>.normalize()</code>	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
<code>.removeAttributeNode()</code>	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
<code>.removeBehavior()</code>	per Methode <code>.addBehavior()</code> einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
<code>.setAttributeNode()</code>	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert

4.3.2.2.12.7. Collectionen des Internet Explorers - Übersicht (englisch)

Hinweis:

Wenn ein HTML-Tag als Zeigernamen in der Punktnotation verwendet wird, dann ist der konkrete Zeiger auf ein Objekt mit Typ laut HTML-Tag gemeint (Objekt erstellbar z.B. per `.createElement(tag_bezeichner_als_string)`)
Das gleiche gilt für Dialog Helper (Objekt `dlgHelper`) etc..

4.3.2.2.12.7.1. Collection .all**Objekte mit der Collection:**

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INS, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, XMP

Syntax:

```
[ collAll = ] object.all
[ oObject = ] object.all(vIndex [, iSubIndex])
```

collAll Array of elements contained by the object.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and

there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

`.length` Sets or retrieves the number of objects in a collection.

Methoden:

`.item` Retrieves an object from the all collection or various other collections.
`.namedItem` Retrieves an object or a collection from the specified collection.
`.tags` Retrieves a collection of objects that have the specified HTML tag name.
`.urns` Retrieves a collection of all objects to which a specified behavior is attached.



zu `.item()`
`oItem = object.item(vIndex [, iSubindex])`

`vIndex` Required. Integer or String that specifies the object or collection to retrieve. If this parameter is an integer, it is the zero-based index of the object. If this parameter is a string, all objects with matching name or id properties are retrieved, and a collection is returned if more than one match is made.

`iSubindex` Optional. Integer that specifies the zero-based index of the object to retrieve when a collection is returned. Returns an object or a collection of objects if successful, or null otherwise.

zu `.namedItem()`
`oItem = object.namedItem(sName)`

`sName` Required. String that specifies the name or id property of the object to retrieve. A collection is returned if more than one match is made. Returns an object or a collection of objects if successful, or null otherwise.

zu `.tags()`
`collElements = object.tags(sTag)`

`sTag` Required. Variant of type String that specifies an HTML tag. It can be any one of the objects exposed by the DHTML Object Model. Returns a collection of element objects if successful, or null otherwise.

zu `.urns()`
`collObjects = object.urns(sUrn)`

`sUrn` Required. String that specifies the behavior's Uniform Resource Name (URN). Returns a collection of objects if successful, or null otherwise.

4.3.2.2.12.7.2. Collection .anchors

Objekte mit der Collection:

document

Syntax:

```
[ collAnchors = ] document.anchors
[ oObject = ] document.anchors(vIndex [, iSubIndex])
```

`collAnchors` Array of a objects.
`oObject` Reference to an individual item in the array of elements contained by the object.
`vIndex` Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.
`iSubIndex` Optional. Position of an element to retrieve. This parameter is used when `vIndex` is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by `iSubIndex`.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.3. Collection .applets

Objekte mit der Collection:

document

Syntax:

```
[ collApplets = ] document.applets
[ oObject = ] document.applets(vIndex [, iSubIndex])
```

`collApplets` Array of applet objects.
`oObject` Reference to an individual item in the array of elements contained by the object.
`vIndex` Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.
`iSubIndex` Optional. Position of an element to retrieve. This parameter is used when `vIndex` is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by `iSubIndex`.



Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.4. Collection .areas**Objekte mit der Collection:**

MAP

Syntax:

```
[ oColl = ] MAP.areas
[ oObject = ] MAP.areas(vIndex [, iSubIndex])
```

oColl Array of area objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

.add() Adds an element to the areas, controlRange, or options collection.

.remove() Removes an element from the collection.

4.3.2.2.12.7.5. Collection .attributes**Objekte mit der Collection:**

A, ABBR, ACRONYM, ADDRESS, APPLET, AREA, attribute, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, XMP

Syntax:

```
[ oColl = ] object.attributes
[ oObject = ] object.attributes(iIndex)
```

oColl Zero-based array of attributes applied to the object.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.getNamedItem() Retrieves an attribute specified with the name property using the attributes collection.

.item() Retrieves an attribute for an element from the attributes collection.

.removeNamedItem() Removes an attribute specified with the name property from an element using the attributes collection.

.setNamedItem() Adds an attribute to an element using an attributes collection.

4.3.2.2.12.7.6. Collection .behaviorUrns**Objekte mit der Collection:**

A, ABBR, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, STYLE, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, WBR, XML, XMP

Syntax:

```
[ oColl = ] object.behaviorUrns
[ oObject = ] object.behaviorUrns(iIndex)
```

oColl Array of URNs identifying the behaviors attached to the element.

oObject Reference to an individual item in the array of elements contained by the object.



iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the behaviorUrns collection.

4.3.2.2.12.7.7. Collection .blockFormats**Objekte mit der Collection:**

dlgHelper

Syntax:

```
[ oColl = ] Dialog Helper.blockFormats
[ oObject = ] Dialog Helper.blockFormats(iIndex)
```

oColl A collection of the names of the available block format tags.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

.Count Retrieves the number of available block format tags.

Methoden:

.Item() Retrieves a string that specifies the name of a block format tag.

4.3.2.2.12.7.8. Collection .boundElements**Objekte mit der Collection:**

event

Syntax:

```
[ oColl = ] event.boundElements
[ oObject = ] event.boundElements(vIndex [, iSubIndex])
```

oColl Array of elements found on a page that are bound to a data set.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

from

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.9. Collection .cells**Objekte mit der Collection:**

TABLE, TR

Syntax:

```
[ oColl = ] object.cells
[ oObject = ] object.cells(vIndex [, iSubIndex])
```

oColl Array of td and th elements contained by the object. If the object is a tr, the array contains elements only in that table row. If

the

object is a table, the array contains all elements in the table.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.10. Collection .childNodes

siehe Objekt Textnode für Text-Knoten als Spezialform von Knoten

Objekte mit der Collection:

A, ACRONYM, ADDRESS, APPLET, AREA, attribute, B, BASE, BASEFONT, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INS, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, OL, OPTION, P, PLAINTEXT, PRE, Q, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, XMP

Syntax:

```
[ oColl = ] object.childNodes
[ oObject = ] object.childNodes(iIndex)
```

oColl Array containing the children of a specified object.
 oObject Reference to an individual item in the array of elements contained by the object.
 iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the childNodes or children collection.
 .urns() Retrieves a collection of all objects to which a specified behavior is attached.

4.3.2.2.12.7.11. Collection .children**Objekte mit der Collection:**

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INS, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, THEAD, TITLE, TR, TT, U, UL, VAR, XMP

Syntax:

```
[ oColl = ] object.children
[ oObject = ] object.children(vIndex [, iSubIndex])
```

oColl Array containing the direct descendants of an object.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the childNodes or children collection.
 .tags() Retrieves a collection of objects that have the specified HTML tag name.
 .urns() Retrieves a collection of all objects to which a specified behavior is attached.

4.3.2.2.12.7.12. Collection .controlRange**Objekte mit der Collection:**

body, selection

Syntax:

Microsoft macht keine Angaben zum Syntax

Vermutung

```
[ oColl = ] object.controlRange
[ oObject = ] object.controlRange(vIndex [, iSubIndex])
```

oColl Array containing the direct descendants of an object.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method

uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.add()	Adds an element to the areas, controlRange, or options collection.
.addElement()	Adds an element to the controlRange collection.
.execCommand()	Executes a command on the current document, current selection, or the given range.
.item()	Retrieves an object from the controlRange collection.
.queryCommandEnabled()	Returns a Boolean value that indicates whether a specified command can be successfully executed
	using execCommand, given the current state of the document.
.queryCommandIndeterm()	Returns a Boolean value that indicates whether the specified command is in the indeterminate state.
.queryCommandState()	Returns a Boolean value that indicates the current state of the command.
.queryCommandSupported()	Returns a Boolean value that indicates whether the current command is supported on the current range.
.queryCommandValue()	Returns the current value of the document, range, or current selection for the given command.
	remove Removes an element from the collection.
.scrollIntoView()	Causes the object to scroll into view, aligning it either at the top or bottom of the window.
	select Makes the selection equal to the current object.

4.3.2.2.12.7.13. Collection .document.all

siehe Collection .all

4.3.2.2.12.7.14. Collection .elements**Objekte mit der Collection:**

FORM

Syntax:

```
[ oColl = ] FORM.elements
[ oObject = ] FORM.elements(vIndex [, iSubIndex])
```

oColl Array of button, input, select, and textArea objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.15. Collection .embeds**Objekte mit der Collection:**

document

Syntax:

```
[ oColl = ] document.embeds
[ oObject = ] document.embeds(vIndex [, iSubIndex])
```

oColl Array of embed objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.16. Collection .filters**Objekte mit der Collection:**

BDO, BODY, BUTTON, CUSTOM, DIV, FIELDSET, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text,



MARQUEE, nextID, RT, RUBY, SPAN, TABLE, TD, TEXTAREA, TH

Syntax:

```
[ oColl = ] object.filters
[ oObject = ] object.filters(vIndex [, iSubIndex])
```

oColl Array of filters applied to the object.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

```
.item()           Retrieves an object from the filters collection or various other collections.
.namedItem()      Retrieves an object or a collection from the specified collection.
```

4.3.2.2.12.7.17. Collection .fonts**Objekte mit der Collection:**

dlgHelper

Syntax:

```
[ oColl = ] Dialog Helper.fonts
[ oObject = ] Dialog Helper.fonts(iIndex)
```

oColl A collection of system-supported fonts.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

.Count Retrieves the number of available block format tags.

Methoden:

```
.Item()           Retrieves a string that specifies the name of a block format tag.
```

4.3.2.2.12.7.18. Collection .forms**Objekte mit der Collection:**

document

Syntax:

```
[ oColl = ] document.forms
[ oObject = ] document.forms(vIndex [, iSubIndex])
```

oColl Array of form objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.19. Collection .frames**Objekte mit der Collection:**

window, document

Syntax:

```
[ oColl = ] object.frames
[ oObject = ] object.frames(vIndex [, iSubIndex])
```

oColl Array of window objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns



the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If
 this
 parameter is a string and there is more than one element with the name or id property equal to the string, the method
 returns
 a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string
 to
 construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this
 collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the filters collection or various other collections.
 .namedItem() Retrieves an object or a collection from the specified collection. 1

4.3.2.2.12.7.20. Collection .images**Objekte mit der Collection:**

document

Syntax:

[oColl =] document.images
 [oObject =] document.images(vIndex [, iSubIndex])

oColl Array of img objects.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method
 returns
 the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If
 this
 parameter is a string and there is more than one element with the name or id property equal to the string, the method
 returns
 a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string
 to
 construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this
 collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.21. Collection .imports**Objekte mit der Collection:**

styleSheet

Syntax:

[oColl =] styleSheet.imports
 [oObject =] styleSheet.imports(iIndex)

oColl Array of imported style sheets.
 oObject Reference to an individual item in the array of elements contained by the object.
 iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the filters collection or various other collections.
 .namedItem() Retrieves an object or a collection from the specified collection.

4.3.2.2.12.7.22. Collection .links**Objekte mit der Collection:**

document

Syntax:

[oColl =] document.links
 [oObject =] document.links(iIndex)

oColl Array of a objects.
 oObject Reference to an individual item in the array of elements contained by the object.
 iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.23. Collection .namespaces**Objekte mit der Collection:**

document

Syntax:

[oNamespace =] document.namespaces



[oObject =] document.namespaces(iIndex)

oNamespace Array of namespace objects.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.add() Creates a new namespace object and adds it to the collection.

.item() Retrieves a namespace object from the namespaces collection.

4.3.2.2.12.7.24. Collection .options

Objekte mit der Collection:

SELECT

Syntax:

[oColl =] SELECT.options

[oObject =] SELECT.options(vIndex [, iSubIndex])

oColl Array of option objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

.add() Adds an element to the areas, controlRange, or options collection.

.remove() Removes an element from the collection.

4.3.2.2.12.7.25. Collection .pages

Objekte mit der Collection:

styleSheet

Syntax:

[oColl =] styleSheet.pages

[oObject =] styleSheet.pages(iIndex)

oColl Array of page objects.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the pages collection.

4.3.2.2.12.7.26. Collection .plugins

Objekte mit der Collection:

clientInformation, navigator

Syntax:

[oColl =] object.plugins

[oObject =] object.plugins(iIndex)

oColl Array that is empty.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the all collection or various other collections.

.namedItem() Retrieves an object or a collection from the specified collection.

.tags() Retrieves a collection of objects that have the specified HTML tag name.

4.3.2.2.12.7.27. Collection .rows

Objekte mit der Collection:

TABLE, TBODY, TFOOT, THEAD

Syntax:

[oColl =] object.rows

[oObject =] object.rows(vIndex [, iSubIndex])



oColl Array of tr objects.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.28. Collection .rules**Objekte mit der Collection:**

styleSheet

Syntax:

```
[ oColl = ] styleSheet.rules
[ oObject = ] styleSheet.rules(iIndex)
```

oColl Array of rules.
 oObject Reference to an individual item in the array of elements contained by the object.
 iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the rules collection.

4.3.2.2.12.7.29. Collection .scripts**Objekte mit der Collection:**

document

Syntax:

```
[ oColl = ] document.scripts
[ oObject = ] document.scripts(vIndex [, iSubIndex])
```

oColl Array of script objects.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.30. Collection .styleSheets

Es gibt auch das Objekt styleSheet (ohne 's' am Ende des Bezeichners).

Objekte mit der Collection:

document

Syntax:

```
[ oColl = ] document.styleSheets
[ oObject = ] document.styleSheets(vIndex [, iSubIndex])
```

oColl Array of styleSheet objects.
 oObject Reference to an individual item in the array of elements contained by the object.
 vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.



a collection of matching elements.
 iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the filters collection or various other collections.
 .namedItem() Retrieves an object or a collection from the specified collection.
 .urns() Retrieves a collection of all objects to which a specified behavior is attached.

4.3.2.2.12.7.31. Collection .tBodies**Objekte mit der Collection:**

TABLE

Syntax:

[oColl =] TABLE.tBodies
 [oObject =] TABLE.tBodies(vIndex [, iSubIndex])

oColl Array of tBody objects.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

wie Collection .all

4.3.2.2.12.7.32. Collection .TextRange**Objekte mit der Collection:**

selection

Syntax:

[oColl =] selection.TextRange
 [oObject =] selection.TextRange(vIndex [, iSubIndex])

oColl Array of elements created with the createRangeCollection method.

oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method

returns

the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If

this

parameter is a string and there is more than one element with the name or id property equal to the string, the method

returns

a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method uses the string

to

construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the filters collection or various other collections.
 .namedItem() Retrieves an object or a collection from the specified collection.

4.3.2.2.12.7.33. Collection .TextRectangle**Objekte mit der Collection:**

TextRectangle,. Achtung: Objekt und Collection haben identischen Bezeichner !

Syntax:

keine Angaben von Microsoft

Vermutung:

[oColl =] selection.TextRange
 [oObject =] selection.TextRange(vIndex [, iSubIndex])

oColl Array of elements created with the createRangeCollection method.



oObject Reference to an individual item in the array of elements contained by the object.

vIndex Required. Integer or string that specifies the element or collection to retrieve. If this parameter is an integer, the method returns the element in the collection at the given position, where the first element has value 0, the second has 1, and so on. If this parameter is a string and there is more than one element with the name or id property equal to the string, the method returns a collection of matching elements.

iSubIndex Optional. Position of an element to retrieve. This parameter is used when vIndex is a string. The method

uses the

string to construct a collection of all elements that have a name or id property equal to the string, and then retrieves from this collection the element at the position specified by iSubIndex.

Eigenschaften:

wie Collection .all

Methoden:

.item() Retrieves an object from the filters collection or various other collections.
 .namedItem() Retrieves an object or a collection from the specified collection.

4.3.2.2.12.8. wichtige Events und ihre Objekte - Übersicht (z.T. in englisch), Methode .fireEvent()

Events bei DOM-Änderung

Wurden Events für window, document, body zugewiesen z.B. für oncontextmenu und werden danach DOM-Änderungen vollzogen, dann können die Events vom Browser während der DOM-Änderungen ignoriert werden, aber am Ende aller DOM-Änderungen wieder aktiv sein.

Grund könnte Ressourcen-Mangel des Browser sein, wenn viele DOM-Änderungen nacheinander folgen. Ausserdem ändert sich oft body.

Mousevents

Das Event onmouseover kann NUR durch die Bewegung der Maus per User ausgelöst werden.

Wenn das Objekt, das onmouseover hat, per Script anhand style.top und style.left verschoben wird und dabei UNTER den unbewegten Mauszeiger gelangt, wird KEIN onmouseover ausgelöst.

Das selbe Prinzip gilt für andere onmouse-Events

Wird die Sanduhr angezeigt, also das Betriebssystem wird aktiv,

dann kann Event erzeugt werden, da die Sanduhr ausserhalb des Dokumentes liegt und der Fokus sich ändert auf Sanduhr
 z.B. mousemove bei IMG wenn cursor über IMG
 mouseout für BODY

Funktion z.B. Eventhandler mit returnValue und mit folgenden Anweisungen im Funktionsrumpf:

Variante 1

return (eval-ausdruck mit return-Anweisung);

eval-ausdruck z.B. eval("return window.event.returnValue");

Es wird innere return-Anweisung von eval abgearbeitet
 vom äusseren return entgegengenommen

Variante 2

eval-ausdruck mit return-Anweisung

eval-ausdruck z.B. eval("return window.event.returnValue");

Es wird innere return-Anweisung von eval abgearbeitet

Ist jetzt z.B. in einem Eventhandler eine äussere return-Anweisung
 zwar notwendig zu kodieren,
 aber nicht kodiert WORDEN,
 dann kann eine Fehlermeldung, dass ein return ausserhalb der
 Funktion liegt

Übersicht zu wichtigen Objekten und Events

Hinweis: bubbles entspricht durchreichen der Events nach oben entlang der DOM-Hierarchie
 cancels entspricht unterbrechbar

.fireEvent()

Fires a specified event on the object.

kann nur innerhalb eines Eventhandlers aktiviert werden, da ansonsten wirkungslos

Syntax

bFired = object.fireEvent(sEvent [, oEventObject])



sEvent Required. String that specifies the name of the event to fire.
 oEventObject Optional. Object that specifies the event object from which to obtain event object properties.

Return Value

Boolean. Returns one of the following values:
 true Event fired successfully.
 false Event was cancelled.

fireEvent() setzt automatisch window.event.cancelBubble auf false, also kein Hochreichen des Events in der DOM-Hierarchie

fireEvent() liefert
 true , wenn
 Event erfolgreich gefeuert
 oder Event nicht feuerebar aber auch nicht cancelbar
 false, wenn
 Event nicht erfolgreich gefeuert
 UND nicht gecancel wurde

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function fnFireEvents()
{
    oDiv.innerText = "The cursor has moved over me!";
    oButton.fireEvent("onclick");
}
</SCRIPT>
</HEAD>
<BODY>
<h1>Using the fireEvent method</h1>
By moving the cursor over the DIV below, the button is clicked.
<P>
<DIV ID="oDiv" onmouseover="fnFireEvents();">
Mouse over this!
</DIV>
<p>
<BUTTON ID="oButton" ONCLICK="this.innerText=I have been clicked!" >
Button</BUTTON>
</BODY>
</HTML>
```

Objekte mit der Methode sind

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, styleSheet, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, WBR, XML, XMP

window-Events

onactivate Fires when the object is set as the active element.
 onafterprint Fires on the object immediately after its associated document prints or previews for printing.
 onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.
 onbeforeprint Fires on the object before its associated document prints or previews for printing.
 onbeforeunload Fires prior to a page being unloaded.
 onblur Fires when the object loses the input focus.
 oncontrolselect Fires when the user is about to make a control selection of the object.
 ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.
 onerror Fires when an error occurs during object loading.
 onfocus Fires when the object receives focus.
 onhelp Fires when the user presses the F1 key while the browser is the active window.
 onload Fires immediately after the browser loads the object.
 onmove Fires when the object moves.
 onmoveend Fires when the object stops moving.
 onmovestart Fires when the object starts to move.
 onresize Fires when the size of the object is about to change.
 onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.
 onresizestart Fires when the user begins to change the dimensions of the object in a control selection.



onscroll Fires when the user repositions the scroll box in the scroll bar on the object.
 onunload Fires immediately before the object is unloaded.

document-Events

onactivate Fires when the object is set as the active element.
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Fires on the source object before the selection is deleted from the document.
 onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.
 onbeforeeditfocus Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.
 onbeforepaste Fires on the target object before the selection is pasted from the system clipboard to the document.
 onclick Fires when the user clicks the left mouse button on the object.
 oncontextmenu Fires when the user clicks the right mouse button in the client area, opening the context menu.
 oncontrolselect Fires when the user is about to make a control selection of the object.
 oncut Fires on the source element when the object or selection is removed from the document and added to the system clipboard.
 ondblclick Fires when the user double-clicks the object.
 ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.
 ondrag Fires on the source object continuously during a drag operation.
 ondragend Fires on the source object when the user releases the mouse at the close of a drag operation.
 ondragenter Fires on the target element when the user drags the object to a valid drop target.
 ondragleave Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.
 ondragover Fires on the target element continuously while the user drags the object over a valid drop target.
 ondragstart Fires on the source object when the user starts to drag a text selection or selected object.
 ondrop Fires on the target object when the mouse button is released during a drag-and-drop operation.
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Fires when the user presses the F1 key while the browser is the active window.
 onkeydown Fires when the user presses a key.
 onkeypress Fires when the user presses an alphanumeric key.
 onkeyup Fires when the user releases a key.
 onmousedown Fires when the user clicks the object with either mouse button.
 onmousemove Fires when the user moves the mouse over the object.
 onmouseout Fires when the user moves the mouse pointer outside the boundaries of the object.
 onmouseover Fires when the user moves the mouse pointer into the object.
 onmouseup Fires when the user releases a mouse button while the mouse is over the object.
 onmousewheel Fires when the wheel button is rotated.
 onmove Fires when the object moves.
 onmoveend Fires when the object stops moving.
 onmovestart Fires when the object starts to move.
 onpaste Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.
 onpropertychange Fires when a property changes on the object.
 onreadystatechange Fires when the state of the object has changed.
 onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.
 onresizestart Fires when the user begins to change the dimensions of the object in a control selection.
 onselectionchange Fires when the selection state of a document changes.
 onstop Fires when the user clicks the Stop button or leaves the Web page.

body-Events

onactivate Fires when the object is set as the active element.
 onafterprint Fires on the object immediately after its associated document prints or previews for printing.
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Fires on the source object before the selection is deleted from the document.
 onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.
 onbeforeeditfocus Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.
 onbeforepaste Fires on the target object before the selection is pasted from the system clipboard to the document.
 onbeforeprint Fires on the object before its associated document prints or previews for printing.
 onbeforeunload Fires prior to a page being unloaded.
 onclick Fires when the user clicks the left mouse button on the object.
 oncontextmenu Fires when the user clicks the right mouse button in the client area, opening the context menu.
 oncontrolselect Fires when the user is about to make a control selection of the object.
 oncut Fires on the source element when the object or selection is removed from the document and added to the system clipboard.
 ondblclick Fires when the user double-clicks the object.
 ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.
 ondrag Fires on the source object continuously during a drag operation.
 ondragend Fires on the source object when the user releases the mouse at the close of a drag operation.
 ondragenter Fires on the target element when the user drags the object to a valid drop target.
 ondragleave Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.
 ondragover Fires on the target element continuously while the user drags the object over a valid drop target.
 ondragstart Fires on the source object when the user starts to drag a text selection or selected object.



ondrop Fires on the target object when the mouse button is released during a drag-and-drop operation.
onfilterchange Fires when a visual filter changes state or completes a transition.
onfocusin Fires for an element just prior to setting focus on that element.
onfocusout Fires for the current element with focus immediately after moving focus to another element.
onkeydown Fires when the user presses a key.
onkeypress Fires when the user presses an alphanumeric key.
onkeyup Fires when the user releases a key.
onload Fires immediately after the browser loads the object.
onlosecapture Fires when the object loses the mouse capture.
onmousedown Fires when the user clicks the object with either mouse button.
onmouseenter Fires when the user moves the mouse pointer into the object.
onmouseleave Fires when the user moves the mouse pointer outside the boundaries of the object.
onmousemove Fires when the user moves the mouse over the object.
onmouseout Fires when the user moves the mouse pointer outside the boundaries of the object.
onmouseover Fires when the user moves the mouse pointer into the object.
onmouseup Fires when the user releases a mouse button while the mouse is over the object.
onmousewheel Fires when the wheel button is rotated.
onmove Fires when the object moves.
onmoveend Fires when the object stops moving.
onmovestart Fires when the object starts to move.
onpaste Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.
onpropertychange Fires when a property changes on the object.
onreadystatechange Fires when the state of the object has changed.
onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.
onresizestart Fires when the user begins to change the dimensions of the object in a control selection.
onscroll Fires when the user repositions the scroll box in the scroll bar on the object.
onselect Fires when the current selection changes.
onselectstart Fires when the object is being selected.
onunload Fires immediately before the object is unloaded.

onabort

Fires when the user aborts the download of an image.
Bubbles No
Cancels Yes
nur Objekt IMG

onactivate

Fires when the object is set as the active element.
Bubbles Yes
Cancels No
Default action Change activation from the event.fromElement to the event.srcElement.
A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onafterprint

Fires on the object immediately after its associated document prints or previews for printing.
Bubbles No
Cancels No
window, BODY, FRAMESET

onafterupdate

Fires on a databound object after successfully updating the associated data in the data source object.
Bubbles Yes
Cancels No
A, BDO, BUTTON, CUSTOM, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT type=radio, INPUT type=text, LABEL, LEGEND, MARQUEE, RT, RUBY, SELECT, SPAN,

TEXTAREA**onbeforeactivate**

Fires immediately before the object is set as the active element.
Bubbles Yes
Cancels Yes
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforecopy

Fires on the source object before the selection is copied to the system clipboard.

Bubbles Yes

Cancels Yes

A, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV,

DL,

DT, EM, FIELDSET, FORM, hn, I, IMG, LABEL, LEGEND, LI, LISTING, MENU, NOBR, OL, P, PLAINTEXT, PRE, S,

SAMP,

SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TD, TEXTAREA, TH, TR, TT, U, UL

onbeforecut

Fires on the source object before the selection is deleted from the document.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforedeactivate

Fires immediately before the activeElement is changed from the current object to another object in the parent document.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

window,

XMP

onbeforeeditfocus

Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.

Bubbles Yes

Cancels Yes

DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, I, INPUT type=button, INPUT

type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE,

MENU,

NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

onbeforepaste

Fires on the target object before the selection is pasted from the system clipboard to the document.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforeprint

Fires on the object before its associated document prints or previews for printing.

Bubbles No

Cancels No

window, BODY, FRAMESET



onbeforeunload

Fires prior to a page being unloaded.
 Bubbles No
 Cancels Yes
 BODY, FRAMESET, window

onbeforeupdate

Fires on a databound object before updating the associated data in the data source object.
 Bubbles Yes
 Cancels Yes
 A, BUTTON, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT type=radio, INPUT type=text, TEXTAREA, LABEL, LEGEND, MARQUEE, SELECT, SPAN, BDO, CUSTOM, RT, RUBY

onblur

Fires when the object loses the input focus.
 Bubbles No
 Cancels No
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password,

INPUT

type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onbounce

Fires when the behavior property of the marquee object is set to "alternate" and the contents of the marquee reach one side of the window.

Bubbles No
 Cancels Yes
 MARQUEE

oncellchange

Fires when data changes in the data provider.
 Bubbles Yes
 Cancels No
 APPLET, BDO, OBJECT

onchange

Fires when the contents of the object or selection have changed.
 Bubbles No
 Cancels Yes
 INPUT type=text, SELECT, TEXTAREA

onclick

Fires when the user clicks the left mouse button on the object.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

oncontextmenu

Fires when the user clicks the right mouse button in the client area, opening the context menu.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU,

nextID,

NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

oncontrolselect

Fires when the user is about to make a control selection of the object.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE,



CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

oncopy
Fires on the source element when the user copies the object or selection, adding it to the system clipboard.
Bubbles Yes
Cancels Yes
A, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, FIELDSET, FORM, hn, HR, I, IMG, LEGEND, LI, LISTING, MENU, NOBR, OL, P, PLAINTEXT, PRE, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TD, TH, TR, TT, U, UL

oncut
Fires on the source element when the object or selection is removed from the document and added to the system clipboard.
Bubbles Yes
Cancels Yes
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondataavailable
Fires periodically as data arrives from data source objects that asynchronously transmit their data.
Bubbles Yes
Cancels No
APPLET, OBJECT, XML

ondatachanged
Fires when the data set exposed by a data source object changes.
Bubbles Yes
Cancels No
APPLET, OBJECT, XML

ondatacomplete
Fires to indicate that all data is available from the data source object.
Bubbles Yes
Cancels No
APPLET, OBJECT, XML

ondblclick
Fires when the user double-clicks the object.
Bubbles Yes
Cancels Yes
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondeactivate
Fires when the activeElement is changed from the current object to another object in the parent document.
Bubbles Yes
Cancels No
A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT



type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

ondrag
Fires on the source object continuously during a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragend
Fires on the source object when the user releases the mouse at the close of a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragenter
Fires on the target element when the user drags the object to a valid drop target.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondragleave
Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragover
Fires on the target element continuously while the user drags the object over a valid drop target.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragstart
Fires on the source object when the user starts to drag a text selection or selected object.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT



type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondrop

Fires on the target object when the mouse button is released during a drag-and-drop operation.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT

type=button,

INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD,

TEXTAREA,

TR, TT, U, UL, VAR, XMP

onerror

Fires when an error occurs during object loading.

Bubbles No

Cancels Yes

IMG, OBJECT, STYLE, window

onerrorupdate

Fires on a databound object when an error occurs while updating the associated data in the data source object.

Bubbles Yes

Cancels No

A, BUTTON, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT type=radio, INPUT type=text, TEXTAREA, LABEL, LEGEND, MARQUEE, SELECT, SPAN, BDO, CUSTOM, RT, RUBY

onfilterchange

Fires when a visual filter changes state or completes a transition.

Bubbles No

Cancels No

BDO, BODY, BUTTON, CUSTOM, DIV, FIELDSET, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, MARQUEE, nextID, RT, RUBY, SPAN, TABLE, TD, TEXTAREA, TH, TR

onfinish

Fires when marquee looping is complete.

Bubbles No

Cancels Yes

MARQUEE

onfocus

Fires when the object receives focus.

Bubbles No

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onfocusin

Fires for an element just prior to setting focus on that element.

Bubbles Yes

Cancels No

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onfocusout

Fires for the current element with focus immediately after moving focus to another element.



Bubbles Yes
 Cancels No
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onhelp
 Fires when the user presses the F1 key while the browser is the active window.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onkeydown
 Fires when the user presses a key.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onkeypress
 Fires when the user presses an alphanumeric key.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onkeyup
 Fires when the user releases a key.
 Bubbles Yes
 Cancels No
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onlayoutcomplete
 Fires when the print or print preview layout process finishes filling the current LayoutRect object with content from the source document.
 Bubbles Yes
 Cancels Yes
 BASE, BASEFONT, BGSOUND, BR, COL, DD, DIV, DL, DT, FONT, HEAD, HR, HTML, HTML Comment, LAYOUTRECT, LI, META, OL, OPTION, P, TITLE, UL

onload
 Fires immediately after the browser loads the object.
 Bubbles No
 Cancels No
 APPLET, BODY, EMBED, FRAME, FRAMESET, IFRAME, IMG, LINK, SCRIPT, window

onlosecapture
 Fires when the object loses the mouse capture.
 Bubbles No
 Cancels No



A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmousedown

Bubbles Yes
Cancels Yes
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY,

TBODY,

TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseenter

Fires when the user moves the mouse pointer into the object.
Bubbles No
Cancels No
A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, HTML, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

XMP

onmouseleave

Fires when the user moves the mouse pointer outside the boundaries of the object.
Bubbles No
Cancels No
A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, HTML, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

XMP

onmousemove

Fires when the user moves the mouse over the object.
Bubbles Yes
Cancels No
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseout

Fires when the user moves the mouse pointer outside the boundaries of the object.
Bubbles Yes
Cancels No
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseover

Fires when the user moves the mouse pointer into the object.
Bubbles Yes
Cancels Yes
A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT



type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseup

Fires when the user releases a mouse button while the mouse is over the object.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmousewheel

Fires when the wheel button is rotated.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmove

Fires when the object moves.

Bubbles Yes

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onmoveend

Fires when the object stops moving.

Bubbles Yes

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

window,

XMP

onmovestart

Fires when the object starts to move.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,



	<p>window, XMP</p>
onpaste	<p>Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.</p> <p>Bubbles Yes</p> <p>Cancels Yes</p> <p>A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP</p>
onpropertychange	<p>Fires when a property changes on the object.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COMMENT, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP</p>
onreadystatechange	<p>Fires when the state of the object has changed.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button,</p>
INPUT	<p>type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, namespace, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, STYLE, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, XML, XMP</p>
onreset	<p>Fires when the user resets a form.</p> <p>Bubbles No</p> <p>Cancels Yes</p> <p>FORM</p>
onresize	<p>Fires when the size of the object is about to change.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>A, ADDRESS, APPLET, B, BIG, BLOCKQUOTE, BUTTON, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL,</p>
DT,	<p>EM, EMBED, FIELDSET, FORM, FRAME, hn, HR, I, IMG, INPUT type=button, INPUT type=file, INPUT type=image,</p>
INPUT	<p>type=password, INPUT type=reset, INPUT type=submit, INPUT type=text, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PRE, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TEXTAREA, TT, U, UL, VAR, window, XMP</p>
onresizeend	<p>Fires when the user finishes changing the dimensions of the object in a control selection.</p> <p>Bubbles Yes</p> <p>Cancels No</p> <p>A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,</p>
I,	<p>IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,</p>
INPUT	<p>type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,</p>
SMALL,	<p>SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP</p>
onresizestart	<p>Fires when the user begins to change the dimensions of the object in a control selection.</p> <p>Bubbles Yes</p>



Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onrowenter

Fires to indicate that the current row has changed in the data source and new data values are available on the object.

Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

onrowexit

Fires just before the data source control changes the current row in the object.

Bubbles No
 Cancels Yes
 APPLET, OBJECT, XML

onrowsdelete

Fires when rows are about to be deleted from the recordset.

Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

onrowsinserted

Fires just after new rows are inserted in the current recordset.

Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

onscroll

Fires when the user repositions the scroll box in the scroll bar on the object.

Bubbles No
 Cancels No
 APPLET, BDO, BODY, CUSTOM, DIV, EMBED, MAP, MARQUEE, OBJECT, TABLE, TEXTAREA, window

onselect

Fires when the current selection changes.

Bubbles No
 Cancels Yes
 BODY, INPUT type=text, TEXTAREA

onselectionchange

Fires when the selection state of a document changes.

Bubbles No
 Cancels No
 document

onselectstart

Fires when the object is being selected.

Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onstart

Fires at the beginning of every loop of the marquee object.

Bubbles No
 Cancels No
 MARQUEE

onstop

Fires when the user clicks the Stop button or leaves the Web page.
 Bubbles No
 Cancels No
 document

onsubmit

Fires when a FORM is about to be submitted.
 Bubbles No
 Cancels Yes
 FORM

ontimeerror

Fires whenever a time-specific error occurs, usually as a result of setting a property to an invalid value.
 Bubbles No
 Cancels No
 A, ACRONYM, ADDRESS, AREA, B, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CODE, DD, DEL, DIR, DIV, DL, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, KBD, LEGEND, LI, LISTING, MARQUEE, MENU, OL, OPTION, P, PLAINTEXT, PRE, Q, S, SAMP, selection, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onunload

Fires immediately before the object is unloaded.
 Bubbles No
 Cancels No
 BODY, FRAMESET, window

4.3.2.2.12.9. wichtige Objekte - Übersicht (z.T. englisch)

Die Informationen basieren auf Angaben von Microsoft. Dabei wurden Angaben z.T. haarsträubend ungenau gemacht (Literaten waren am Werk):

Beispiel

'getBoundingClientRect Retrieves an object that specifies the bounds of a collection of TextRectangle objects.'
 übersetzt: Erhalten eines Objektes, das die Grenzen von einer Collection aus TextRange-Objekten angibt.

Beispiel: Knoten steht für Objekt, das selbst Objekte also Knoten enthalten kann.

Knoten ist aus Sicht der Dokument-Hierarchie:
 Objekt ist aus Sicht des DOM.
 Objekt-Knoten ist also doppeltgemoppelt sinnlos

Übersetzungen ins Deutsche sind nur teilweise erfolgt.

Attribute sind im HTML-Kontext hinterlegt, wobei diese im Script-Kontext Eigenschaften heißen.

HTML-Attribute werden in Grossbuchstaben angegeben
 Eigenschaften werden in Gross-Kleinbuchstaben angegeben.

Die Implementation von Attributen und Eigenschaften obliegt ausnahmslos Microsoft, wobei die Kompatibilität zu Webstandards im HTML und CSS und zu anderen Browsern mehr oder weniger vollzogen wird: Objektspezifisch.

Es kann mehr Eigenschaften geben als Attribute.

Es gibt Attribute, die sind nicht als Eigenschaft hinterlegt (also per Punktnotation nicht ansprechbar)

Es fehlen teilweise sinnvolle Attribute bzw. Eigenschaften.

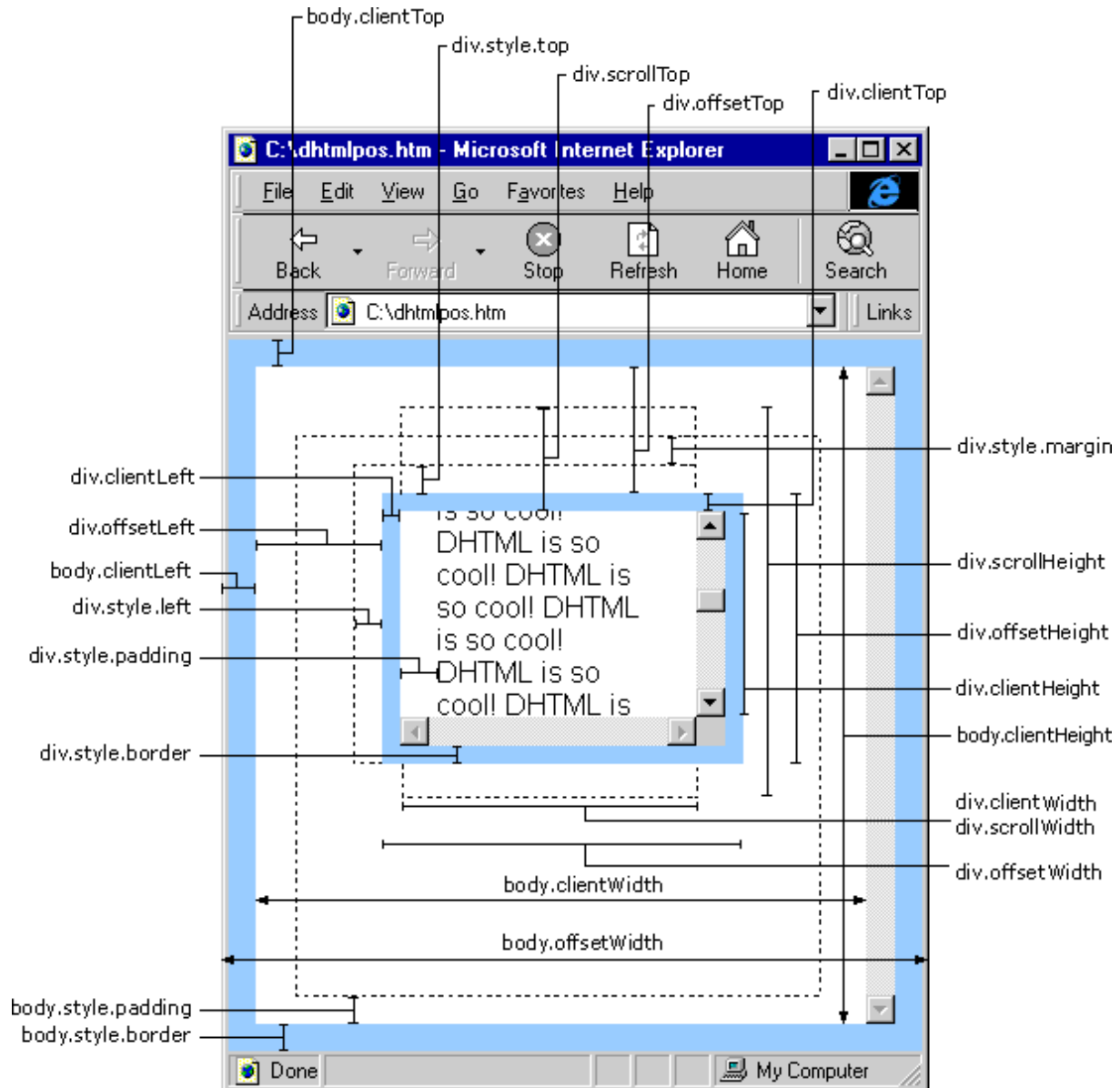
Of benötigte kombinierte Angaben aus Style sind nicht als Style-Eigenschaft hinterlegt, sondern müssen durch den Programmierer ermittelt und aktuell gehalten werden.

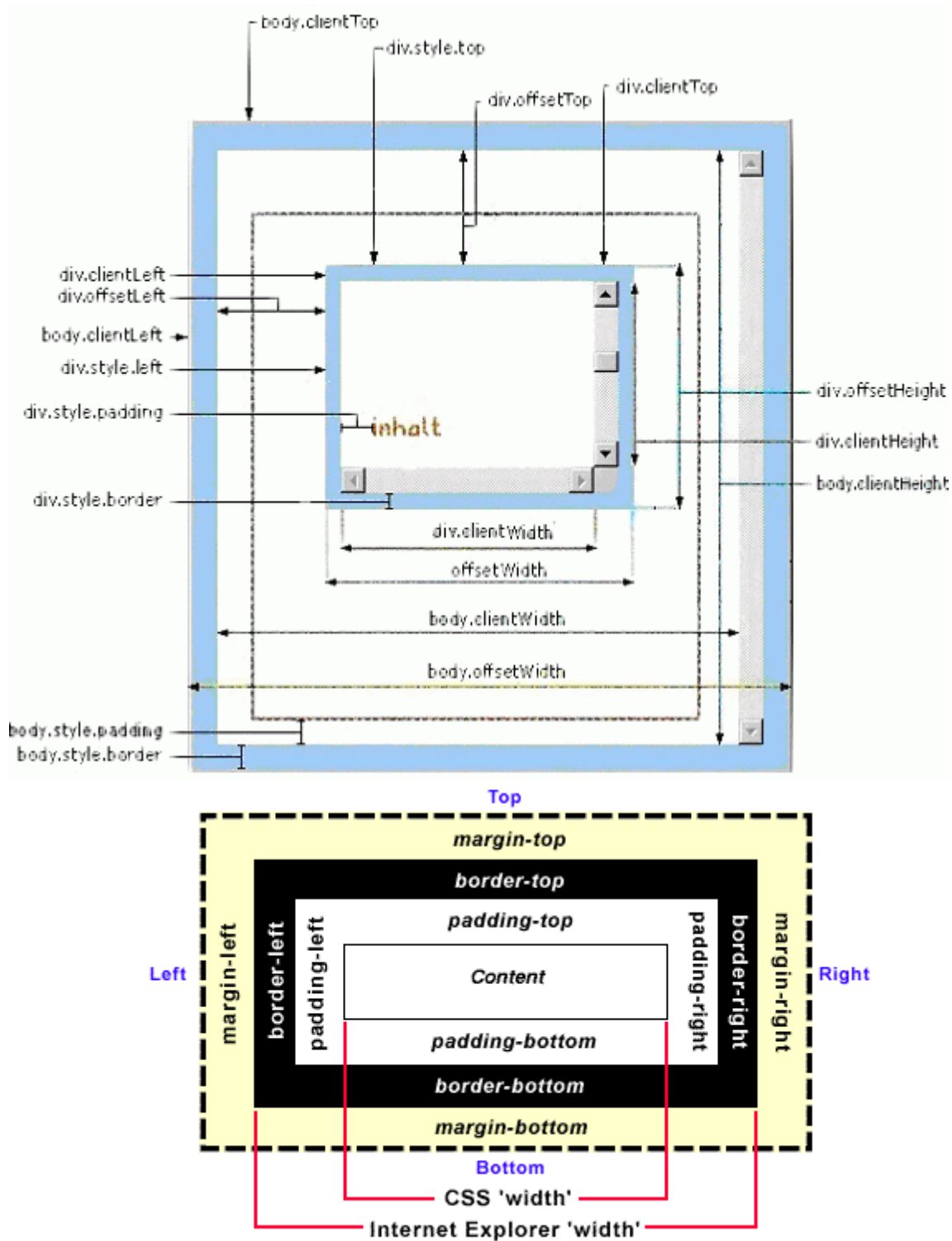
Die Zuweisung eines Wertes zu einer Eigenschaft kann per Ergibtzeichen '=' oder mit einer speziellen Funktion (Methode) erfolgen.
 (Für Style-Werte gibt es noch zusätzliche Ergibtzeichen.)

Die Ermittlung bzw. das Setzen von Eigenschaften eines Objektes kann anhand Eigenschaften und Methoden erfolgen (wobei Prototyping nicht ausgeschlossen wird, das aber nicht zwingend Browseraktivitäten auslösen muss).

Styles und CSS beim IE







Folgende Vererbung muss beachtet werden:

Objekt window -> Objekt document -> Objekt HEAD
 Objekt BODY -> diverse HTML-Elemente
 -> Objekt event für eventauslösende Objekte, allerdings nicht für jedes (z.B. popup)

Die Vererbung gibt normalerweise die Art der Punktnotation vor:

z.B. document.body
 window.event

Achtung: window ist selbst eine Instanz, wenn per window.open() erzeugt.

Objekt window ist Eltern-Objekt von

clientInformation Contains information about the Web browser.

clipboardData Provides access to predefined clipboard formats for use in editing operations.



document Represents the HTML document in a given browser window.
 of event Represents the state of an event, such as the element in which the event occurred, the state of the keyboard keys, the location the mouse, and the state of the mouse buttons.
 external Allows access to an additional object model provided by host applications of the Microsoft® Internet Explorer browser components.
 history Contains information about the URLs visited by the client.
 location Contains information about the current URL.
 navigator Contains information about the Web browser.
 screen Contains information about the client's screen and rendering capabilities.

Objekt document ist Eltern-Objekt von

BODY body Specifies the beginning and end of the document body.
 implementation Contains information about the modules supported by the object.
 location Contains information about the current URL.
 selection Represents the active selection, which is a highlighted block of text, and/or other elements in the document on which a user or a script can carry out some action.
 TITLE title Contains the title of the document.

Objekt Body ist Eltern-Objekt von

currentStyle Represents the cascaded format and style of the object as specified by global style sheets, inline styles, and HTML attributes.
 runtimeStyle Represents the cascaded format and style of the object that overrides the format and style specified in global style sheets, inline styles, and HTML attributes.
 style Represents the current settings of all possible inline styles for a given element.

Es werden nur einige Objekte nachfolgend skizziert.

4.3.2.2.12.9.1. A

window.status und HREF-Attribut

Der IE erzeugt bei mouseover über <A> mit HREF z.B. auf Datei eine Änderung von window.status
 mit mouseover wird HREF-Wert angezeigt
 mit mouseout wird window.status auf den Wert VOR mouseover gesetzt
 bei rechten Maustasteklick die Anzeige des HREF-Wertes solange, wie rechte Maustaste gedrückt bleibt.

Abhilfe dafür schafft return true; für
 onmouseover
 onfocus da oncontextmenu nicht die Anzeige des HREF per rechter Maus unterdrückt

Beispiel:

```
<a href="link.htm" onmouseover="window.status='Text mit Mouseover';return true"
onfocus="window.status='Text mit Mouseover';return true"
onmouseout="window.status='Standardtext vor mouseover';return true">
das ist der link-Text
</a>
```

oncontextmenü und HREF-Attribut

Das Sperren von oncontextmenu unterdrückt nicht die Anzeige des HREF-Attributwertes: Es muss onfocus-Event behandelt werden.

Aktiver BGSound wird gestoppt, wenn TARGET-Attribut eines Link (z.B. A) nicht angegeben wurde und damit '_self' als Standardwert gilt.

Port eines HREF anzeigen

```
<SCRIPT>
function getPort()
{alert ("FTP: " + oFtp.port + "\n" + "HTTP: " + oHttp.port); }
</SCRIPT>

<A HREF="ftp://www.microsoft.com" onclick="getPort();" ID=oFtp>ftp</A>
<A HREF="http://www.microsoft.com" onclick="getPort();" ID=oHttp>http</A>
```

HREF-Wert als String mit

Internet Explorer 4 und 5	2083	Zeichen
6	508	Zeichen

HREF-Wert mit JScript:

Bsp.: ...

xxxxxx mit Anzahl Zeichen		
Internet Explorer 4 und 5	2083	Zeichen
6	508	Zeichen



Bsp.:

gepackter Quellcode:

```
<BODY>
<a href="javascript:function C(v){return '<td>'
+v+'</td><td>'+((v>>4).toString(16)+(v&15).toString(16)).toUpperCase()+'</td><td bgcolor=DDDDDD><b>&'+'#
+v+'</b></td>';} var c=4,b=Math.ceil(224/c),a='<table border=0><tr>';for(j=0;j<c;j++){
a+='<td>DEC</td><td>HEX</td><td><b>ASC</b></td>';}a+='</tr>';for(i=33;i<33+b;i++){a+='<tr>';
for(j=0;j<c;j++){t=i+(j*b);if(t<=255)a+=C(t);}a+='</tr>';}a+='</table>';var W=open("",'width=500,height=600,
left=0,top=0,resizable,scrollbars');W.document.writeln(a);">Klick</a>
<BODY>
```

aufbereiteter Quellcode

```
<BODY>
<a href="javascript: function C(v)
{
return
'<td>'
+v+'</td><td>'
+((v>>4).toString(16)+(v&15).toString(16)).toUpperCase()
+'</td><td bgcolor=DDDDDD><b>&'
+'#
+v
+'</b></td>';
}
var c=4,b=Math.ceil(224/c),a='<table border=0><tr>';
for(j=0;j<c;j++)
{
a+='<td>DEC</td><td>HEX</td><td><b>ASC</b></td>';
}
a+='</tr>';
for(i=33;i<33+b;i++)
{
a+='<tr>';
for(j=0;j<c;j++)
{
t=i+(j*b);if(t<=255)a+=C(t);
}
a+='</tr>';
}
a+='</table>';
var W=open("",'width=500,height=600,left=0,top=0,resizable,scrollbars');
W.document.writeln(a);">Klick</a>
<BODY>
```

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 charset Zeichensatz zum Encodieren des Objektes ermitteln bzw. setzen
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
 COORDS coords Koordinaten des Objektes ermitteln bzw. setzen
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen
 END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 hash Subsection im HREF-Attribut nach dem Ziffernzeichen # ermitteln bzw. setzen
 hasMedia Status (boolean) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 host Hostname mit Portnummer einer Url oder Location ermitteln bzw. setzen
 hostname Hostname ohne Port einer Url oder Location ermitteln bzw. setzen
 HREF href Ankerzielpunkt ermitteln bzw. setzen
 HREFLANG hreflang Sprachencode des Objektes ermitteln bzw. setzen
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)



innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 METHODS Liste aller HTTP-Methoden ermitteln bzw. setzen, die vom Objekt unterstützt werden
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nameProp Dateiname aus dem Wert von HREF oder SRC ermitteln
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 pathname Dateiname aus dem Pfad des Objektes ermitteln bzw. setzen
 port Portnummer aus der Url ermitteln bzw. für Url setzen
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 protocol Protokoll aus einer Url ermitteln bzw. für Url setzen
 readyState Status des Objektes ermitteln
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 REL rel Beziehungen zwischen Objekt und Linkziel ermitteln bzw. setzen
 REV rel Beziehungen zwischen Objekt und Linkziel ermitteln bzw. setzen
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 search String hinter '?' ermitteln bzw. setzen
 SHAPE shape Umriss des Objektes ermitteln bzw. setzen
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
 systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TARGET target Zielfenster bzw. Ziel-Frame ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type MIME-Typ des Objektes ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 URN urn Uniform Resource Name (URN) für das Zieldokument ermitteln bzw. setzen

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt



oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
 onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Event erzeugt, wenn Mausrad gedreht wird
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen



removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:active :active Layout für benutzen Link
 :hover :hover Layout nach mouseover über Link
 :link :link Layout für bisher noch nicht geklickten Link
 :visited :visited Layout für bereits benutzten Link
 ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen



margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
 overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
 overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 text-decorationBlink Textblinken ein aus
 text-decorationLineThrough Text durchstreichen ein aus
 text-decorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 text-decorationOverline Linie über Text zeichnen ein aus
 text-decorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 width width Breite eines Objektes ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.2. BGSOUND

BGSound-Objekt mit DOM erzeugen:

```
// Soundobjekt erzeugen
var BGSoundObjekt=document.createElement('<BGSOUND LOOP="0" VOLUME="-10">');
document.body.insertBefore(BGSoundObjekt); // in DOM einbinden
```

// Es muss der BODY als Objekt erkannt sein !

Es können mehrere BGSOUND-Objekte pro Webseite erzeugt werden, wobei zu beachten ist, dass pro BGSOUND-Objekt der Browser Ressourcen bereitstellen muss.

Vorladen von Sound in den Browsercache:

analog zu IMG (siehe dort) geht nicht, da es keine JScript-Funktion zu Sound gibt.

Die Erzeugung eines eigenen Konstruktor für new macht wenig Sinn, wenn der Konstruktor den Browser nicht anweisen kann, Daten zu laden (im Gegensatz zu new Image() siehe dort .src)
 Wieso Microsoft solchen Konstruktor in JScript vergessen hat, ist unklar. Zumal BGSOUND ohne Plugins, also ohne EMBED etc. auskommt und direkt in Windows vorhandene Codecs nutzt.

kann anhand Test-BGSOUND-Objekt erfolgen (erzeugen per createElement() etc.), wobei Volume auf -10 gesetzt sein muss, damit kein Sound hörbar wird während des Vorladens. Vorladen erfolgt mit Belegung von .src. Nach dem Vorladen

kann

das Testobjekt per document.body.removeChild(BGSoundObjekt); aus dem DOM entfernt werden.

readyState kann nach .src-Zuweisung genutzt werden, um festzustellen, wann ein Sound geladen ist bzw. beginnt zu erklingen (um dann per Rekursion in der Sounddauer den Verstummungszeitpunkt abzufangen). Auch wenn readyState den Status 'complete' liefert, kann es nachfolgende Probleme geben, die auch readyState beeinflussen könnten).

Sounddateien benötigen meist wegen Dateiumfang längere Ladezeiten (siehe oben Vorladen), wobei

Internet lahmen kann,
 Server des Webseiten-Hosters langsam sind,
 geringe Internetzugriffsrate vorliegt (z.B. ISDN anstelle DSL),
 Virens Scanner die Dateien zeitintensiv verarbeiten könnten.

Sound unter Windows wird nicht zwingend in Echtzeit wiedergegeben, da Windows kein Echtzeitbetriebssystem ist.



Das Ruckeln von Sound kann z.B. an überlasteten Timern von Windows liegen (besonders wenn Timer in JScript intensiv genutzt werden). Viele Schleifen in JScript benötigen ebenfalls mehr Echtzeit-Ressourcen.

Sound unter Windows kann gnadenlos abgewürgt werden, wenn das Desktop-Symbol geklickt wird. Aktiver BGSound wird gestoppt, wenn TARGET-Attribut eines Link (z.B. A) nicht angegeben wurde und damit '_self' als Standardwert gilt.

Wird Sound unter Windows vollständig oder teilweise abgewürgt, dann kann JScript, das Sound erzeugt hat, trotzdem weiterlaufen (inklusive Timer, die dann nicht mehr synchron zum erklingenden Sound sind).

Soundwiedergabe unter Windows kann auch unter Nutzung von Media Player erfolgen, so dass BGSOUND nicht zwingend nötig ist. Der Media Player unterliegt aber regelmäßigen Änderungen (im Gegensatz zu BGSOUND).

Echtzeit-Animation von Sound ist per Timeline-Konzept des IE möglich, aber nicht gerade einfach zu programmieren und eventuell auch Änderungen je nach Browserversion unterliegend.

Sound beim IE und Netscape:

```
<html>
<head>
<title>Text des Titels</title>
<!-- Microsoft: -->
<bgsound src="background.mid" loop="infinite">
</head>
<body>
<!-- Netscape: -->
<embed src="background.mid" autostart="true" loop="true" hidden="true" height="0" width="0">
<h1>Inhalt der Seite</h1>
</body>
</html>
```

Attribute

BALANCE balance Balance ermitteln bzw. setzen (nur Stereo)
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
LOOP loop Anzahl der lückenlosen Wiederholungen der Wiedergabe des Sounds (unendlich möglich)
nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
nodeName Name des Teiltyps des Knoten ermitteln
nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
nodeValue Wert eines Knoten ermitteln bzw. setzen
outerHTML HTML-Kontext um das Objekt ermitteln bzw. setzen
outerText Text-Kontext um das Objekt ermitteln bzw. setzen
parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
readyState Status des Objektes ermitteln
scopeName Namensbereich des Objektes ermitteln
sourceIndex Index des Objektes in der Collection document.all ermitteln
SRC src Url der Sounddatei (wenn Leerkette so BGSOUND stummgeschaltet)
tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
tagUm Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
VOLUME volume Volume (-10 bis 0, -10 ist stumm, man beachte Windows-Lautstärke-Regler)

Events

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben
onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
applyElement Das Objekt als Eltern- oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
dragDrop Drag-Event für das Objekt erzeugen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern



getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 insertAdjacentElement Nachbar-Objekt einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsformierung für Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen

4.3.2.2.12.9.3. BODY

Wenn Fenster ohne Scrollleisten erzeugen:

```
<BODY STYLE="overflow:visible">
```

Der BODY ist per Style nicht immer mehr veränderbar

z.B. wird folgendes im IE innerhalb onload akzeptiert:

```
document.body.style.background="url('h.jpg') white center no-repeat fixed"
allerdings die einzelne Stylezuweisung per document.body.style .....
nicht unbedingt!
```

Beispiel für Scroll-Leisten ein aus:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function Hidden() {
document.body.style.overflow='hidden';
}
function Show() {
document.body.style.overflow="";
}
//-->
</script>
<a href="#" onclick="Hidden()">Scrollleisten verstecken</a><br>
<a href="#" onclick="Show()">Scrollleisten wieder anzeigen</a>
```

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
 ALINK aLink Farbe des aktiven Links ermitteln bzw. setzen
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BACKGROUND background Hintergrundbild ermitteln bzw. setzen
 BGCOLOR bgColor Diese Eigenschaft ist nicht mehr verwendbar !
 BGPROPERTIES bgProperties Hintergrundbild-Eigenschaften ermitteln bzw. setzen
 blockDirection Richtung des Block-Element-Flusses (links nach rechts oder umgekehrt) ermitteln bzw. setzen
 BOTTOMMARGIN bottomMargin Bottom-margin des Body ermitteln bzw. setzen
 canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
 DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 LEFTMARGIN leftMargin Left-margin des Body ermitteln bzw. setzen
 LINK link Farbe eines Links ermitteln bzw. setzen
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln



nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 NOWRAP noWrap automatischer Zeilenumbruch ein aus
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 RIGHTMARGIN rightMargin Right-margin für Body ermitteln bzw. setzen
 scopeName Namensbereich des Objektes ermitteln
 SCROLL scroll Scrollbarverfügbarkeit generell ein aus
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TEXT text Textfarbe ermitteln bzw. setzen
 t:TIMESTARTRULE timeStartRule Startpunkt der Timeline ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TOPMARGIN topMargin Top-margin ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VLINK vLink Farbe eines benutzten Links ermitteln bzw. setzen

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterprint Event erzeugt direkt nach Ende des Druckes bzw. Druckvorschau zum Objekt
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeprint Event erzeugt direkt vor Beginn des Druckes bzw. Druckvorschau zum Objekt
 onbeforeunload Event erzeugt direkt vor Entladen des Dokumentes
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
 onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt



onmousewheel Event erzeugt, wenn Mausrad gedreht wird
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresizing Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onselect Event erzeugt wenn die Selektion des Objektes geändert wird
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 onunload Event erzeugt, wenn Objekt entladen ist (aus Browser entladen)

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 createControlRange controlRange-Collection von Nicht-Text-Elementen erzeugen
 createTextRange TextRange-Objekt erzeugen
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 doScroll Erzeugt Klick auf die Scrollbar des Objektes
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 pause Pausen the timeline on the HTML document.
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 resume Resumes Eine pausierende Timeline fortsetzen (Pause beenden)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-letter :first-letter Layout der ersten Buchstaben im Objekt festlegen
 :first-line :first-line Layout der ersten Zeile im Objekt festlegen
 ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen



background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
border-left borderLeft Linker Rahmen
border-left-color borderLeftColor Linker Rahmen
border-left-style borderLeftStyle Linker Rahmen
border-left-width borderLeftWidth Linker Rahmen
border-right borderRight Rechter Rahmen
border-right-color borderRightColor Rechter Rahmen
border-right-style borderRightStyle Rechter Rahmen
border-right-width borderRightWidth Rechter Rahmen
border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
direction direction Leserichtung ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
pixelBottom Bottom-Position der unteren Kante des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
scrollbar-3dlight-color scrollbar3dLightColor Farbe der oberen und linken Kante von Scroobar und Scroll-Pfeile ermitteln bzw. setzen
scrollbar-arrow-color scrollbarArrowColor Farbe der Scrol-Pfeile ermitteln bzw. setzen
scrollbar-base-color scrollbarBaseColor Farbe der Hauptelement der Scrollbar, Scrollbox und Scrollpfeile ermitteln bzw. setzen
scrollbar-darkshadow-color scrollbarDarkShadowColor Schattenfarbe der Scrollbar ermitteln bzw. setzen



scrollbar-face-color scrollbarFaceColor Farbe der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-highlight-color scrollbarHighlightColor Farbe der oberen und linken Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-shadow-color scrollbarShadowColor Farbe der unteren und der rechten Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-track-color scrollbarTrackColor Farbe der Trackelemente der Scrollbar ermitteln bzw. setzen
 text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-align-last textAlignLast Ausrichtung der letzten Text-Linie im Objekt ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
 text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-kashida-space textKashidaSpace Layout kashida ermitteln bzw. setzen
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können
 word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

Collectionen

all Zeiger auf Zeigerfeld aller inneren Objekte
 attributes Zeiger auf Zeigerfeld aller Attribute-Objekte des Objektes
 behaviorUrns Zeiger auf Stringfeld von Uniform Resource Name (URN) der attachierten Behavior
 childNodes Zeiger auf Zeigerfeld aller HTML-Elemente und Text-Knoten (TextNodes) des Objektes
 children Zeiger auf Zeigerfeld aller DHTML-Objekte des Objektes
 filters Zeiger auf Zeigerfeld aller zum Objekt vorhandenen Filter
 timeAll Zeiger auf Zeigerfeld aller getimten Objekte
 timeChildren Zeiger auf Zeigerfeld aller getimten Top-Level-Kinder des Objektes

4.3.2.2.12.9.4. clientInformation

Informationen zum Browser, siehe auch Objekt navigator und userProfile

Attribute

appCodeName Codename des Browsers
 appMinorVersion Unterversion des Browsers
 appName Anwendungsname des Browsers
 appVersion Betriebssystemplattform und Version des Browsers
 browserLanguage Sprache im Browser als Anwendung
 cookieEnabled Status ob Cookies dauerhaft ermöglicht werden oder nun temporär
 cpuClass Bezeichner der Prozessor-Klasse (CPU class)
 onLine Status ob System global on- oder offline
 platform Betriebssystemplattform des Users
 systemLanguage Standardsprache des Betriebssystems
 userAgent HTTP-User-Agent
 userLanguage Sprache im System

Events

keine

Methoden

javaEnabled Ermitteln ob Java verfügbar ist
 taintEnabled Ermitteln ob data tainting möglich ist

Styles

keine

Objekte

userProfile Objekt zur Verwaltung (auch Speichern) von Userdaten im Datenbereich des Internet Explorers

Collectionen

plugins Collection aller eingebetteten EMBED-Objekte im Dokument
 Diese Collection existiert nur zum Zweck der Komaptibilität mit Nicht-IE-Browsern, da der IE in der Regel EMBED nicht nutzt: Anstelle von Plugins werden z.B. Active-X verwendet.
 Stellt ein Plugin-Anbieter kein Active-X-Control bereits, so muss das Plugin verwendet werden.
 Syntax

[oColl =] object.plugins

[oObject =] object.plugins(iIndex)

oColl Array that is empty.

oObject Reference to an individual item in the array of elements contained by the object.

iIndex Required. Integer that specifies the zero-based index of the item to be returned.

Attribut: length Sets or retrieves the number of objects in a collection.



Methoden

item()	Retrieves an object from the all collection or various other collections.
namedItem()	Retrieves an object or a collection from the specified collection.
tags()	Retrieves a collection of objects that have the specified HTML tag name.

4.3.2.2.12.9.5. DIV

Es kann sein, dass DIV ein automatisches
 rendert.

Sollte einem Objekt eine eigenschaft fehlen, die der DIV aber hat, dann das Objekt im DIV-Container erzeugen:

z.B. hat IMG im Style zwar height ABER KEIN width ...(warum auch immer dieser Schwachsinn)

Für Textformatierungen spezieller Art z.B. SPAN oder FONT verwenden.

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen

ALIGN align Ausrichtung des Objektes beim Rendern

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen

blockDirection Richtung des Block-Element-Flusses (links nach rechts oder umgekehrt) ermitteln bzw. setzen

canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen

DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist

DATAFORMATAS dataFormatAs Art der Renders von Daten im Objekt ermitteln bzw. setzen

DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen

END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

hasMedia Status (booleen) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist

HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

NOWRAP noWrap automatischer Zeilenumbruch ein aus

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Status des Objektes ermitteln

recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen

scrollWidth Scrollbreite vom Objekt ermitteln

sourceIndex Index des Objektes in der Collection document.all ermitteln

STYLE style inline-Style des Objektes setzen

SYNCMaster syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss

systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln



systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der User-Einstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltip (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Selektierbarkeit des Objektes ausschalten

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Fokus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Fokus erhält (und damit auch aktiv wird)
 onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
 onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Event erzeugt, wenn Mausrad gedreht wird
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Größe (Dimension) des Objektes ändert
 onresizend Event erzeugt, wenn die Änderung der Größe (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Größe (Dimension) des Objektes beginnt
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren



blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt ein onclick Ereignis erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingetragene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 doScroll Erzeugt Klick auf die Scrollbar des Objektes
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Dokuments vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-letter :first-letter Layout der ersten Buchstaben im Objekt festlegen
 :first-line :first-line Layout der ersten Zeile im Objekt festlegen
 ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links rechts oben unten ermitteln bzw. setzen



bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
 overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
 overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
 page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
 pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 scrollbar-3dlight-color scrollbar3dLightColor Farbe der oberen und linken Kante von Scroobar und Scroll-Pfeile ermitteln bzw. setzen
 scrollbar-arrow-color scrollbarArrowColor Farbe der Scrol-Pfeile ermitteln bzw. setzen
 scrollbar-base-color scrollbarBaseColor Farbe der Hauptelement der Scrollbar, Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-darkshadow-color scrollbarDarkShadowColor Schattenfarbe der Scrollbar ermitteln bzw. setzen
 scrollbar-face-color scrollbarFaceColor Farbe der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-highlight-color scrollbarHighlightColor Farbe der oberen und linken Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-shadow-color scrollbarShadowColor Farbe der unteren und der rechten Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-track-color scrollbarTrackColor Farbe der Trackelemente der Scrollbar ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-align-last textAlignLast Ausrihtung der letzten Text-Linie im Objekt ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen



text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-kashida-space textKashidaSpace Layout kashida ermitteln bzw. setzen
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können
 width width Breite eines Objektes ermitteln bzw. setzen
 word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.6. document

document.title kann nicht durch (Folge von) Leerzeichen gelöscht werden. Es müssen Zeichen ungleich Leerzeichen enthalten sein !

Läuft ein Dokument im Frameset, so

ist der sichtbare Titel im Fensterkopf DER vom FRAMESET und nicht vom Frame.

muss parent.document.title verwendet werden, um von Frame aus den sichtbaren Titel vom FRAMSET manipulieren zu können.

Attribute

activeElement Retrieves the object that has the focus when the parent document has focus.
 alinkColor Sets or retrieves the color of all active links in the document.
 bgColor Deprecated. Sets or retrieves a value that indicates the background color behind the object.
 charset Sets or retrieves the character set used to encode the object.
 compatMode .. siehe tiefer
 cookie Sets or retrieves the string value of a cookie.
 defaultCharset Retrieves the default character set from the current regional language settings
 designMode Sets or retrieves a value that indicates whether the document can be edited.
 dir Sets or retrieves a value that indicates the reading order of the object.
 doctype Retrieves the document type declaration associated with the current document.
 documentElement Retrieves a reference to the root node of the document.
 domain Sets or retrieves the security domain of the document.
 expando Sets or retrieves a value indicating whether arbitrary variables can be created within the object.
 fgColor Sets or retrieves the foreground (text) color of the document.
 fileCreatedDate Retrieves the date the file was created.
 fileModifiedDate Retrieves the date the file was last modified.
 fileSize Retrieves the file size.
 implementation Retrieves the implementation object of the current document.
 lastModified Retrieves the date the page was last modified, if the page supplies one.
 linkColor Sets or retrieves the color of the document links.
 parentWindow Retrieves a reference to the container object of the window.
 protocol Sets or retrieves the protocol portion of a URL.
 readyState Retrieves a value that indicates the current state of the object.
 referrer Retrieves the URL of the location that referred the user to the current page.
 uniqueID Retrieves an autogenerated, unique identifier for the object.
 URL Sets or retrieves the URL for the current document.
 URLUnencoded Retrieves the URL for the document, stripped of any character encoding.
 vlinkColor Sets or retrieves the color of the links that the user has visited.
 XMLDocument Retrieves a reference to the XML Document Object Model (DOM) exposed by the object.
 XSLDocument Retrieves a reference to the top-level node of the XSL document.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschneitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert



ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
onkeydown Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
onkeydown Event erzeugt, sobald eine gedrückte Taste losgelassen wird
onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
onmouseover Event erzeugt, sobald Maus das Objekt betritt
onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
onmousewheel Event erzeugt, wenn Mausrad gedreht wird
onmove Event erzeugt, wenn Objekt bewegt wird
onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
onselectionchange Fires when the selection state of a document changes.
onstop Fires when the user clicks the Stop button or leaves the Web page.

Methoden

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
clear Not currently supported.
close Closes an output stream and forces the sent data to display.
createAttribute Creates an attribute object with a specified name.
createComment Creates a comment object with the specified data.
createDocumentFragment Creates a new document.
createElement Creates an instance of the element for the specified tag.
createEventObject Generates an event object for passing event context information when using the fireEvent method.
createStyleSheet Creates a style sheet for the document.
createTextNode Creates a text string from the specified value.
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
elementFromPoint Returns the element for the specified x and y coordinates.
execCommand Executes a command on the current document, current selection, or the given range.
focus Dem Objekt den Fokus geben (und das Objekt damit aktiv setzen)
getElementById Returns a reference to the first object with the specified value of the ID attribute.
getElementsByName Retrieves a collection of objects based on the value of the NAME attribute.
getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
hasFocus Retrieves the value indicating whether the object currently has focus.
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
open This method works in two ways. It opens a document to collect the output of the write and writeln methods. In this case, only the first two parameters, url and name are used. When values for the additional parameters are specified, this method opens a window in the same way as the window.open method for the window object.
queryCommandEnabled Returns a Boolean value that indicates whether a specified command can be successfully executed using execCommand, given the current state of the document.
queryCommandIndeterm Returns a Boolean value that indicates whether the specified command is in the indeterminate state.
queryCommandState Returns a Boolean value that indicates the current state of the command.
queryCommandSupported Returns a Boolean value that indicates whether the current command is supported on the current range.
queryCommandValue Returns the current value of the document, range, or current selection for the given command.
recalc Recalculates all dynamic properties in the current document.
releaseCapture Maus-Event-Überwachung zum Objekt entfernen
setActive Objekt aktiv setzen, aber nicht Fokus geben
write Writes one or more HTML expressions to a document in the specified window.
writeln Writes one or more HTML expressions, followed by a carriage return, to a document in the specified window.

Styles

compatMode Sets or retrieves a value that indicates whether standards-compliant mode is switched on for the object.

Collectionen

all Zeiger auf Zeigerfeld aller inneren Objekte
anchors Retrieves a collection of all a objects that have a name and/or id property. Objects in this collection are in HTML source order.
applets Retrieves a collection of all applet objects in the document.
childNodes Zeiger auf Zeigerfeld aller HTML-Elemente und Text-Knoten (TextNodes) des Objektes
embeds Retrieves a collection of all embed objects in the document.
forms Retrieves a collection, in source order, of all form objects in the document.
frames Retrieves a collection of all window objects defined by the given document or defined by the document associated with the given window.
images Retrieves a collection, in source order, of img objects in the document.
links Retrieves a collection of all a objects that specify the HREF property and all area objects in the document.
namespaces Retrieves a collection of namespace objects.
scripts Retrieves a collection of all script objects in the document.
styleSheets Retrieves a collection of styleSheet objects representing the style sheets that correspond to each instance of a link or style object in the document.



4.3.2.2.12.9.7. event

Ein Objekt hat nur bestimmte Events implementiert bekommen.

Events eines aller Objekte werden über das im window vorhandene Event-Objekt verwaltet. Daher muss window.event kodiert werden.

.attachEvent() Eventhandler zuweisen und Überwachung des Events durch den Handler aktivieren
belegt NICHT das Event mit dem den Eventhandler (also keine Zuweisung der Funktion auf das Event), sondern steuert

anhand des Eventhandlers das Event.
anstelle von onXXXX="..." im HTML-Tag
Achtung: NUR beim Parsen wird onXXXX in die Eventsteuerung des Dokumentes eingebunden
Es geht theoretisch auch folgende Variante:

objekt.eventbezeichner=eventhandlerbezeichner; // ohne ()

prüfen Wenn mehrere Handler für ein und dasselbe Event, so Aufruf der Handler in Zufallsfolge, es sei denn, die Handler selbst die Aufruffolge (ist zu programmieren)

Syntax

bSuccess = object.attachEvent(sEvent, fpNotify)

sEvent String mit Name des standard DHTML Events.

fpNotify Zeiger auf Eventhandler

bSuccess true Event wird überwacht
false Eventzuordnung nicht möglich, also Event wird nicht überwacht

Der Rückkercode sollte einer Variablen zugewiesen werden.

detachEvent() für Stop der Überwachung

eine Zuweisung per objekt.onXXXX=zeiger_auf_funktion; direct nach attachEvent()

füllt sehrwohl onXXXX, so dass alert(objekt.onXXXX); NICHT null anzeigt
ändert nichts am attach

Beispiel:

```
mySpan.attachEvent("onmouseover", ReaktionBeiMouseOver);
mySpan.onmouseover=zeiger;
```

attach bleibt aktiv

und die Zuweisung mySpan.onmouseover=zeiger;

wird vollzogen, gelangt jedoch nicht oder verzögert in die
Eventsteuerung des Browsers

Achtung: wenn doch wirksam, so wird HREF wieder wirksam, falls
mySpan.onmouseover nicht auch HREF-Anzeige sperrt !

.cancelBubble

wenn true so Event nicht in der DOM-Hierarchie nach oben durchgereicht
nicht von Eltern-Elementen auswertbar

DOM-Hierarchie: oben nach unten

window -> document -> body -> Elemente des body

event ist Objekt von window, also window.event, und muss in window sowie dessen Kinder wie document benutzt werden

window ist also der zentrale Event-Punkt und leitet Events also zu und von Kindern weiter

Eventfolge ist also immer aufsteigend bis window per Script kontrollierbar

eine absteigende Eventfolge gibt es NUR intern und kann nicht per Script kontrolliert werden (intern wegen DOM-Hierarchie)

Ist ein Event im window-Objekt per handler gesperrt (z.B. Handler liefert false), dann ist das Event auch im Kind gesperrt

Wenn dann das Kind einen eigenen Eventhandler zum identischen Event hat,

kann dieser per cancelBubble verhindern, dass das Event z.B.

zu windows gelangt, wobei window.event generell zu nutzen ist.

Das ist wichtig, damit die Eventsperre in window nicht aktiv wird.

Das Event kommt von einem Kind von window, geht zu window, würde

dort gesperrt zurück zum Kind gehen, wenn cancelBubble false wäre

Es ist daher zu prüfen, ob cancelBubble zum Event standardgemäß true oder false ist.

Wenn window.event ausgelesen wird, dann ist zu beachten:

In einer Eventfolge, die durch die Kinder von window ausgelöst wird,

wobei Kinder parallel Events auslösen können,

hat window.event Werte in Echtzeit.

Das ist ein riesen Problem: Das Auslesen von window.event muss also in Echtzeit erfolgen ... und das bei Script,
das über einen Interpreter, der selbst Laufzeit braucht (z.B. Parser) abgearbeitet wird.

Beispiel zum auslesen von window.event

```
if(window.event!=null) // JETZT das letzte Event genau 1x ermitteln in Echtzeit,
                        // also alles direkt nacheinander
{
  X01=window.event;
  if(X01.srcElement!=null)
```



```

{ X02=X01.srcElement;
  d010[0]=X02;
  if(X02.tagName!=null){ d010[1]=X02.tagName;}
  d010[2]=X01.returnValue;
  d010[3]=X01.type;
  d010[4]=X01.cancelBubble;
  d010[5]=X01.reason;

  // Zeiger auf Objekt das das Event ausgelöst hat
  // Tagname
  // Event-Returnwert
  // Eventname ohne 'on', Beispiel 'click' und nicht 'onclick'
  // false so Event in der Eventhierarchie durchgereicht
  //      0      Data-Übertragung erfolgreich
  //      1      Data-Übertragung abgebrochen
  //      2      Data-Übertragung fehlerhaft (alles
  //              ausser 0 und 1)
  // Bezeicher des Attributes, das mit Eventeintritt geändert wurde

  d010[6]=X01.propertyName;
  X00=true;
}
}

```

.fireEvent()

Beispiel für Event erzeugen: Man beachte das Retten von cancelBubble WEGEN fireEvent(), das cancelBubble immer auf false setzt, also das Hochreichen erlaubt !!!!

```

if(X02)
{ X03=window.event.cancelBubble; // wegen fireEvent() muss cancelBubble gerettet werden
  X04=document.createEventObject(); // lokales Eventobjekt erzeugen, das Event in X00 feuert
  X02=X00.fireEvent(X01,X04); // Event in X00 erzeugen und damit den Eventhandler von X00 aktivieren
                                // Achtung: fireEvent setzt automatisch window.event.cancelBubble auf false,
                                //
                                //          kein Hochreichen des Events in der DOM-Hierarchie
                                // fireEvent() liefert
                                //      true , wenn   Event erfolgreich gefeuert
                                //          oder Event nicht feuerebar aber auch nicht cancelbar
                                //      false, wenn Event nicht erfolgreich gefeuert UND nicht gecancelt
                                //
/
wurde
window.event.cancelBubble=X03; // gerettetes cancelBubble holen
                                // wenn true war, so wird Event in der DOM-Hierarchie auch hochgereicht
}

```

fireEvent() muss von einen Eventhandler aus aktiviert werden, da window.event im Zugriff sein muss.
Hinweis: Sobald im Programmcode window.event kodiert und window.event nicht vorhanden, wird Scriptfehler erzeugt.

fireEvent

Fires a specified event on the object.

Syntax

bFired = object.fireEvent(sEvent [, oEventObject])

Parameters

sEvent Required. String that specifies the name of the event to fire.
oEventObject Optional. Object that specifies the event object from which to obtain event object properties.
erzeugt mit document.createEventObject()

Return Value

Boolean. Returns one of the following values:

true Event fired successfully.
false Event was cancelled, falls nicht cancelbar so true geliefert

Achtung: Es werden automatisch für window.event gesetzt:

cancelBubble	auf false
returnValue	auf true damit Event auch ausgelöst wird
srcElement	Zeiger vom Objekt, das das Event feuert
type	Name vom Event, das gefeuert wurde also sEvent

Objekte mit .fireEvent()

nicht window, window.document aber dafür window.document.body
nicht window.event
A, ABBR, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT



```

type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT
type=reset,
INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK,
LISTING, MAP, MARQUEE, MENU, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT,
OL,
OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN,
STRIKE, STRONG, styleSheet, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH,
THEAD, TITLE, TR, TT, U, UL, VAR, WBR, XML, XMP

```

Eventhandler per createElement() erzeugen

```

function test1(...){...}
function test2(...){...}

// Karo-HTML-Code für createElement ermitteln
X13='<DIV';
X13+= ' onmouseover="test1(';
X13+=X12; // Parameterliste mit Kommatrennung
X13+= ' onmousedown="test2(';
X13+=X12; // Parameterliste mit Kommatrennung
X13+= '></DIV>';
// Karo als DIV mit Inhalt und Eventhandler erzeugen
X16=document.createElement(X13);X00=(X16!=null);
if(X00){X17=document.body.appendChild(X16);X00=(X17!=null);} // Eventhandler werden attachiert
if(X00){X00=(X16==X17);}
if(X00){.....}

```

Attribute

Abstract Retrieves the Abstract content of the entry banner in an Advanced Stream Redirector (ASX) file using the event object.

altKey Sets or retrieves a value that indicates the state of the ALT key.

altLeft Sets or retrieves a value that indicates the state of the left ALT key.

Banner Retrieves the Banner content of an entry in an ASX file using the event object.

button Sets or retrieves the mouse button pressed by the user.

cancelBubble Sets or retrieves whether the current event should bubble up the hierarchy of event handlers.

clientX Sets or retrieves the x-coordinate of the mouse pointer's position relative to the client area of the window, excluding window decorations and scroll bars.

clientY Sets or retrieves the y-coordinate of the mouse pointer's position relative to the client area of the window, excluding window decorations and scroll bars.

contentOverflow Retrieves a value that indicates whether the document contains additional content after processing the current LayoutRect object.

ctrlKey Sets or retrieves the state of the CTRL key.

ctrlLeft Sets or retrieves the state of the left CTRL key.

dataFld Sets or retrieves the data column affected by the oncellchange event.

fromElement fromElement Sets or retrieves the object from which activation or the mouse pointer is exiting during the event.

keyCode Sets or retrieves the Unicode key code associated with the key that caused the event.

MoreInfo Retrieves the MoreInfo content of an entry banner in an ASX file through the event object.

nextPage Retrieves the position of the next page within a print template.

offsetX Sets or retrieves the x-coordinate of the mouse pointer's position relative to the object firing the event.

offsetY Sets or retrieves the y-coordinate of the mouse pointer's position relative to the object firing the event.

propertyName Sets or retrieves the name of the property that changes on the object.

qualifier Sets or retrieves the name of the data member provided by a data source object.

reason Sets or retrieves the result of the data transfer for a data source object.

recordset Sets or retrieves from a data source object a reference to the default record set.

repeat Retrieves whether the onkeydown event is being repeated.

returnValue Sets or retrieves the return value from the event.

saveType Retrieves the clipboard type when oncontentsave fires.

screenX Retrieves the x-coordinate of the mouse pointer's position relative to the user's screen.

screenY Sets or retrieves the y-coordinate of the mouse pointer's position relative to the user's screen.

shiftKey Sets or retrieves the state of the SHIFT key.

shiftLeft Retrieves the state of the left SHIFT key.

srcElement Sets or retrieves the object that fired the event.

srcElement ist ausschliesslich für Objekt event.

liefert Zeiger auf das Event auslösende Objekt

Zu diesem Objekt können dann die zum Objekt die gültigen Attribute etc. verwendet werden z.B tagName

srcFilter Sets or retrieves the filter object that caused the onfilterchange event to fire.

srcUrn Retrieves the Uniform Resource Name (URN) of the behavior that fired the event.

toElement Sets or retrieves a reference to the object toward which the user is moving the mouse pointer.

type Sets or retrieves the event name from the event object.

userName Retrieves the sFriendlyName parameter that is passed to the useService method.

wheelDelta Sets or retrieves the distance and direction the wheel button has rolled.

x Sets or retrieves the x-coordinate, in pixels, of the mouse pointer's position relative to a relatively positioned parent element.

y Sets or retrieves the y-coordinate, in pixels, of the mouse pointer's position relative to a relatively positioned parent element.

Events

je nach Objekt

Methoden

keine



Styles

keine

Collectionen

bookmarks Returns a collection of Microsoft ActiveX® Data Objects (ADO) bookmarks tied to the rows affected by the current event.

boundElements Returns a collection of all elements on the page bound to a data set.

4.3.2.2.12.9.8. FONT**Attribute**

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen

canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CLASS className Klassenname des Objektes ermitteln bzw. setzen

COLOR color Sets or retrieves the color to be used by the object.

CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen

END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen

FACE face Sets or retrieves the current typeface family.

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

hasMedia Status (boolean) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Status des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

SIZE size Sets or retrieves the font size of the object.

sourceIndex Index des Objektes in der Collection document.all ermitteln

STYLE style inline-Style des Objektes setzen

SYNCMaster syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss

systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln

systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird

systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung

systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind

TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen

uniqueID vom Browser intern erzeugt ID des Objektes ermittel (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

UNSELECTABLE Selektierbarkeit des Objektes ausschalten

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird

onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird

onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird

onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird

onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält

onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird

onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)



onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert

ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird

onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)

onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält

onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält

onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren

onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird

onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird

onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben

onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert

onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt

onmouseout Event erzeugt, sobald die Maus das Objekt verlässt

onmouseover Event erzeugt, sobald Maus das Objekt betritt

onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt

onmousewheel Event erzeugt, wenn Mausrad gedreht wird

onmove Event erzeugt, wenn Objekt bewegt wird

onmoveend Event erzeugt, wenn Bewegung eines Objektes endet

onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt

onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden

onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet

onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt

onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln

getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

getExpression Ausdruck für eine Eigenschaft liefern

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

insertAdjacentHTML HTML-Code als Nachbar-Code einfügen

insertAdjacentText Text als Nachbar-Text einfügen

insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten

releaseCapture Maus-Event-Überwachung zum Objekt entfernen

removeAttribute Attribute aus einem Objekt entfernen



removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

ACCELERATOR accelerator Tastaturkürzel setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
 overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
 overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen



posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können
 width width Breite eines Objektes ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.9. FRAME

Attribute

ALLOWTRANSPARENCY allowTransparency Sets or retrieves whether the object can be transparent.
 APPLICATION Indicates whether the content of the object is an HTML Application (HTA) and, therefore, exempt from the browser security model.
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BORDERCOLOR borderColor Sets or retrieves the border color of the object.
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 contentWindow Retrieves the window object of the specified frame or iframe.
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 FRAMEBORDER frameBorder Sets or retrieves whether to display a border for the frame.
 HEIGHT height Retrieves the height of the object.
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 longDesc Sets or retrieves a Uniform Resource Identifier (URI) to a long description of the object.
 MARGINHEIGHT marginHeight Sets or retrieves the top and bottom margin heights before displaying the text in a frame.
 MARGINWIDTH marginWidth Sets or retrieves the left and right margin widths before displaying the text in a frame.
 NAME name Sets or retrieves the frame name.
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 NORESIZE noResize Sets or retrieves whether the user can resize the frame.
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 SCROLLING scrolling Sets or retrieves whether the frame can be scrolled.
 SECURITY Sets the value indicating whether the source file of a frame or iframe has specific security restrictions applied.
 self Retrieves a reference to the current window or frame.
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 SRC src Sets or retrieves a URL to be loaded by the object.
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 WIDTH width Retrieves the width of the object.

Events



onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 oncontrolsselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status soehe readyState)
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Fires when the user begins to change the dimensions of the object in a control selection.

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 height height Objekthöhe ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen



layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.10. FRAMESET

Attribute

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
BORDER border Sets or retrieves the space between the frames, including the 3-D border.
BORDERCOLOR borderColor Sets or retrieves the border color of the object.
canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
CLASS className Klassenname des Objektes ermitteln bzw. setzen
COLS cols Sets or retrieves the frame widths of the object.
firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
FRAMEBORDER frameBorder Sets or retrieves whether to display a border for the frame.
FRAMESPACING frameSpacing Sets or retrieves the amount of additional space between the frames.
HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
innerHTML Retrieves the HTML between the start and end tags of the object.
isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
LANG lang zu benutzende Sprache ermitteln bzw. setzen
LANGUAGE language Scriptsprache ermitteln bzw. setzen
lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
NAME name Sets or retrieves the frame name.
nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
nodeName Name des Teiltypen des Knoten ermitteln
nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
nodeValue Wert eines Knoten ermitteln bzw. setzen
outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
readyState Status des Objektes ermitteln
ROWS rows Sets or retrieves the frame heights of the object.
scopeName Namensbereich des Objektes ermitteln
sourceIndex Index des Objektes in der Collection document.all ermitteln
TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
onafterprint Event erzeugt direkt nach Ende des Druckes bzw. Druckvorschau zum Objekt
onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
onbeforeprint Event erzeugt direkt vor Beginn des Druckes bzw. Druckvorschau zum Objekt
onbeforeunload Event erzeugt direkt vor Entladen des Dokumentes
onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
oncontrolsselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird



onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onunload Event erzeugt, wenn Objekt entladen ist (aus Browser entladen)

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-left borderLeft Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-top borderTop Oberer Rahmen
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 height height Objekthöhe ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
 pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.11. HEAD

Kopf des Dokumentes als Container also <HEAD> .. </HEAD>
 liegt hinter <HTML> und vor <BODY>



Folgende HTML-Tags können im HEAD hinterlegt werden

BASE
BASEFONT
BGSOUND
LINK
META
nextID
SCRIPT
STYLE
TITLE

Attribute

canHaveChildren Retrieves a value indicating whether the object can contain children.
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
CLASS className Klassenname des Objektes ermitteln bzw. setzen
firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
LANG lang zu benutzende Sprache ermitteln bzw. setzen
lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
nodeName Name des Teiltypen des Knoten ermitteln
nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
nodeValue Wert eines Knoten ermitteln bzw. setzen
ownerDocument Retrieves the document object associated with the node.
parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
PROFILE profile Sets or retrieves one or more Uniform Resource Identifier (URI)(s) in which the object's properties and legal values for those properties are defined.
readyState Status des Objektes ermitteln
scopeName Namensbereich des Objektes ermitteln
scrollHeight Scrollhöhe vom Objekt ermitteln
scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
scrollWidth Scrollbreite vom Objekt ermitteln
sourceIndex Index des Objektes in der Collection document.all ermitteln
tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

Events

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
contains Ermitteln ob Objekt andere Objekte enthält
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
dragDrop Drag-Event für das Objekt erzeugen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
getAdjacentText Benachbarten Textstring liefern
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
insertAdjacentElement Nachbar-Objekt einfügen
insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
removeAttribute Attribute aus einem Objekt entfernen
removeAttributeNode Attribut aus Objekt entfernen
removeBehavior Behavior aus einem Objekt entfernen
removeChild Kindknoten aus einem Objekt entfernen
removeNode Objektknoten aus Dokument-Hierarchie entfernen



replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
 pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.12. HTML

Dokument als HTML-Container für HTML-Elemente des Dokumentes also <HTML> .. </HTML>
 zugleich Container für HEAD und BODY (HEAD vor BODY)

Attribute

canHaveChildren Retrieves a value indicating whether the object can contain children.
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML Retrieves the HTML between the start and end tags of the object.
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontext um das Objekt ermitteln bzw. setzen
 outerText Text-Kontext um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 SCROLL scroll Scrollbarverfügbarkeit generell ein aus
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 VERSION version Sets or retrieves the Document Type Definition (DTD) version that governs the current document.
 XMLNS Declares a namespace for custom tags in an HTML document.



Events

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten

removeAttribute Attribute aus einem Objekt entfernen

removeAttributeNode Attribut aus Objekt entfernen

removeBehavior Behavior aus einem Objekt entfernen

removeChild Kindknoten aus einem Objekt entfernen

removeNode Objektknoten aus Dokument-Hierarchie entfernen

replaceAdjacentText Benachbarten Text ersetzen

replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen

replaceNode Knoten durch anderen Knoten ersetzen

setAttribute Wert eines Attributes setzen

setAttributeNode Attribut-Knoten in Objekt einfügen

swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.

:hover :hover Sets the style of an element when the user hovers the mouse pointer over it.

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen

color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)

cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen

display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)

font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen

font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen

font-size fontSize Höhe des Fonts ermitteln bzw. setzen

font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen

font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)

fontWeight Sets or retrieves the numeric weight of the font of the object.

font-weight fontWeight Font-Wichtung ermitteln bzw. setzen

letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen

line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen

overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf

overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf

overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf

text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen

text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen

textDecorationBlink Textblinken ein aus

textDecorationLineThrough Text durchstreichen ein aus

textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus

textDecorationOverline Linie über Text zeichnen ein aus

textDecorationUnderline Linie unter Text zeichnen ein aus

text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen

text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen

visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)

word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen



word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus

4.3.2.2.12.9.13. HTML-Kommentar

Bsp.: <!-- Das ist ein HTML-Kommentar, der weder ausgewertet noch gerendert wird -->

Attribute

innerHTML Sets or retrieves the HTML between the start and end tags of the object.

outerHTML Sets or retrieves the object and its content in HTML.

tagName Retrieves the tag name of the object.

text Retrieves or sets the text of the object as a string.

Events

onlayoutcomplete Fires when the print or print preview layout process finishes filling the current LayoutRect object with content from the source document.

Methoden

keine

Styles

keine

4.3.2.2.12.9.14. IMG

IMG hat im Style zwar .height ABER KEIN .width.

Wird ein Bild per createElement() mit Dimension Höhe == Breite == 0 erzeugt, so ist das Bild eventuell nicht renderbar und jede nachträgliche Änderung der Bild-Dimension per JScript wird auch nicht gerendert. (Bild-Dimension > 0 lässt Bild rendern.)

Ab IE 5 werden mindestens folgende Dateiformate unterstützt

- avi - Audio-Visual Interleaved (AVI)
- bmp - Windows Bitmap (BMP)
- emf - Windows Enhanced Metafile (EMF)
- gif - Graphics Interchange Format (GIF)
- jpg, .jpeg - Joint Photographic Experts Group (JPEG)
- mov - Apple QuickTime Movie (MOV)
- mpg, .mpeg - Motion Picture Experts Group (MPEG)
- png - Portable Network Graphics (PNG)
- wmf - Windows Metafile (WMF)
- xbm - X Bitmap (XBM)

Dateiformate müssen nicht zwingend browserabwärtskompatibel gerendert werden.

Als SRC kann auch eine Quelle wie FTP genutzt werden.

Als unsichtbares Bild, das als Platzhalter dienen kann (auch z.B. gekachelt) empfiehlt sich Gif mit 1x1 Pixeln transparent. Dieses Bild ist schwer entdeckbar, so dass Events wie z.B. onmouseover nicht auffallen.

Vorladen von Bildern:

Wenn SRC-Belegung erfolgen soll, dann kann die Bilddatei bereits im Browser-Cache liegen, wobei der Browser merkt, ob die Datei schon gecacht ist, also diese nicht doppelt anfordert. Diese Verfügbarkeit der Daten im Browsercache nennt man vorladen. Das Vorladen funktioniert aber nur, wenn die Webseite im Internet-Zugriff läuft, z.B. per HTTP-Server von Apache als virtual Host auf der lokalen Festplatte. Der Browser cacht nicht, wenn die Webseite direkt von der lokalen Festplatte gestartet wird, ohne im Internetzugriff zu sein. Grund: Es wird direkt von Festplatte gelesen ohne den HTTP-Server dazwischen. Der Browsercache wird verwaltet je nach Browsereinstellungen: Werden Daten geändert, so müssen diese auch gecacht werden. Wenn der Browser Daten nur am Dateinamen erkennt, aber nicht inhaltlich, dann werden geänderte Daten bei gleichem Dateinamen nicht neu gecacht. Ergo muss der User den Browser-Cache löschen. Man beachte auch Meta-Tag-Angaben.

```
var BildObjektZumVorladen=new Image (...); // IMG-Objekt per JScript-Funktion erzeugen
BildObjektZumVorladen.src='test.bmp'; // vorladen in den Browsercache
...
// .... Bildobjekt per createElement() und appendChild erzeugen als var BildZeiger
...
BildZeiger.src= BildObjektZumVorladen.src; // gecachte Daten zuweisen wenn benötigt
```

Hinweis: .readyState ist für IMG anwendbar, so dass ermittelt werden kann, wann eine Datei vorgeladen ist.

Bilddateien benötigen ev. wegen Dateiumfang längere Ladezeiten (siehe oben Vorladen), wobei Internet lahmen kann, Server des Webseiten-Hosters langsam sind, geringe Internetzugriffsrate vorliegt (z.B. ISDN anstelle DSL), Virens Scanner die Dateien zeitintensiv verarbeiten könnten.

zu new Image():

```
var Zeiger=new Image(Breite,Hoehe);
```



```
// Breite und Höhe in Pixel optional (entweder beide oder keiner)
// erzeugt IMG-Objekt, das WIE das HTML-Element IMG ist
// ohne ein HTML-Element zu sein
// Grund: HTML-Elemente müssen per createElement() und append.... erzeugt werden, wobei
// der Zeiger auf das HTML-Objekt geliefert wird UND das HTML-Objekt im DOM
// hinterlegt wird
// new Image() erzeugt keine DOM-Änderung

// Achtung: Es ist möglich, einen Zeiger eines HTML-Objektes mit dem aus new Image() zu überschreiben,
// was aber nicht garantiert, dass dann das "neue" HTML-Objekt auch noch im DOM ist,
// da new Image() eine JavaScript-Funktion ist (der Scriptmaschine) und keine Funktion
// im DOM.

// Es ist möglich Zeiger_auf_IMG_per_new_Image.src zu nutzen, um den .src eines HTML-Objektes
// zu füllen, wobei aber Zeiger_auf_IMG_per_new_Image nicht gelöscht werden darf.

// per new Image() erzeugtes IMG-Objekt hat u.a.
// .style
// .readyState
```

Beispiel für Vorladen mit readyState

```
function test()
{if(X02.readyState=='complete')
{alert('complete');} // alert erfolgt !
else
{X01=window.setTimeout('test()',200);}
}

var X00='t1.jpg'; // grosse Datei mit mehreren MBytes verwenden!
var X01=0; // test(): TimeoutID
X02=new Image(737,1100); // IMG erzeugen mit Breite und Höhe
X02.src=X00; // Laden der Bilddaten
// alert(X02.readyState!=null); // true
// alert(X02.readyState=='complete'); // false
test(); // readyState rekursiv abklappern
```

Dateigrösse eines Bildes ermitteln per . fileSize

```
var DateiGrosseNumerischInBytes=zeiger_auf_img.fileSize;
```

Abschalten des GalleryImage-Symbols (bei mouseover über Bild) per.galleryImg

```
zeiger_auf_img. galleryImg=wert;

wert      true oder 'yes'      Symbol erscheint
          false oder 'no'      ist Standard
          Symbol erscheint nicht
```

Tipp: Collection aller IMG-Zeigerer mitteln und dann diese per Schleife abklappern, um .gallerImg zu setzen.
Den IMG-Zeiger aus createElement() in ein globales Feld speichern, das als Collection dient.

Bild innerhalb einer USEMAP oder ISMAP:
.galleryImage überschreibt mapping-Attribut

Zeitpunkt für Start eines Video festlegen
per zeiger_auf_img.start [= sStart]

```
sStart      'fileopen' ist Standard      Start wenn Datei geladen wurde
            'mouseover'                  Start wenn Datei geladen wurde UND mouseover erfolgt ist
```

Es werden nur Video-Daten akzeptiert, deren Url per .dynsrc und nicht per .src zugewiesen wurde.

```
<BODY>
<IMG ID="TEST">
<SCRIPT>
TEST.dynsrc='test.avi';
TEST.start='mouseover';
</SCRIPT>
</BODY>
```

siehe auch INPUT TYPE='image'

Attribute



ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ALIGN align Sets or retrieves how the object is aligned with adjacent text.
 ALT alt Sets or retrieves a text alternative to the graphic.
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
 BORDER border Sets or retrieves the width of the border to draw around the object.
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 complete Retrieves whether the object is fully loaded.
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
 DYNsrc dynsrc Sets or retrieves the address of a video clip or VRML world to display in the window.
 END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
 fileCreatedDate Retrieves the date the file was created.
 fileModifiedDate Retrieves the date the file was last modified.
 fileSize Retrieves the file size.
 fileUpdatedDate Retrieves the date the file was last updated.
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 GALLERYIMG galleryImg Sets or retrieves whether the My Pictures image toolbar is visible for the current image.
 hasMedia Status (boolean) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist
 Height height Objekthöhe ermitteln bzw. setzen
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 HSPACE hspace Sets or retrieves the horizontal margin for the object.
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 ISMAP isMap Sets or retrieves whether the image is a server-side image map.
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEditable Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 longDesc Sets or retrieves a Uniform Resource Identifier (URI) to a long description of the object.
 LOOP loop Anzahl der lückenlosen Wiederholungen der Wiederabgabe des Sounds (unendlich möglich)
 LOWSRC lowsrc Sets or retrieves a lower resolution image to display.
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nameProp Dateiname aus dem Wert von HREF oder SRC ermitteln
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 protocol Protokoll aus einer Url ermitteln bzw. für Url setzen
 readyState Retrieves a value that indicates the current state of the object.
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 SRC src Sets or retrieves a URL to be loaded by the object.
 start start Sets or retrieves when a video clip file should begin playing.
 STYLE style inline-Style des Objektes setzen
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss



systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der User-Einstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 USEMAP useMap Sets or retrieves the URL, often with a bookmark extension (#name), to use as a client-side image map.
 VSPACE vspace Sets or retrieves the vertical margin for the object.
 WIDTH width Sets or retrieves the calculated width of the object.

Events

onabort Event wird erzeugt, wenn das Laden des Bildes abgebrochen wurde
 onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerror Fires when an error occurs during object loading.
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
 onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Event erzeugt, wenn Mausexplorer gedreht wird
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren



blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen



font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.
visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.15. Input

Diese Objekt ist Basisobjekt für Input-Objekt mit Attribut Type.

Zeitpunkt für Start eines Video festlegen

per zeiger_auf_input.start [= sStart]

sStart	'fileopen' ist Standard
	'mouseover'

Start wenn Datei geladen wurde

Start wenn Datei geladen wurde UND mouseover erfolgt ist

Es werden nur Video-Daten akzeptiert, deren Url per .dynsrc und nicht per .src zugewiesen wurde.

```
<BODY>
<INPUT TYPE="image" ID="TEST">
<SCRIPT>
TEST.dynsrc='test.avi';
TEST.start='mouseover';
</SCRIPT>
</BODY>
```

siehe auch IMG

Attribute

ACCEPT accept Sets or retrieves a comma-separated list of content types.

ALIGN align Sets or retrieves how the object is aligned with adjacent text.

ALT alt Sets or retrieves a text alternative to the graphic.

complete Retrieves whether the object is fully loaded.

DYNSRC dynsrc Sets or retrieves the address of a video clip or VRML world to display in the window.

HSPACE hspace Sets or retrieves the horizontal margin for the object.

LOOP loop Anzahl der lückenlosen Wiederholungen der Wiedergeb des Sounds (unendlich möglich)

LOWSRC lowsrc Sets or retrieves a lower resolution image to display.



start start Sets or retrieves when a video clip file should begin playing.
 USEMAP useMap Sets or retrieves the URL, often with a bookmark extension (#name), to use as a client-side image map.
 VALUE value Sets or retrieves the default or selected value of the control.
 VSPACE vspace Sets or retrieves the vertical margin for the object.

Events

keine

Methoden

keine

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 layout-flow layoutFlow Flussrichtung und Flussart des Kontext des Objektes ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.16. Input button**Attribute**

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATAFORMATAS dataFormatAs Art der Renders von Daten im Objekt ermitteln bzw. setzen
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 defaultValue Sets or retrieves the initial contents of the object.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 form Retrieves a reference to the form that the object is embedded in.
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 SIZE size Sets or retrieves the size of the control.
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)



tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type Retrieves or initially sets the type of input control represented by the object.
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALUE value Sets or retrieves the default or selected value of the control.
 WIDTH width Sets or retrieves the calculated width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 createTextRange TextRange-Objekt erzeugen
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen



fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 select Highlights the input area of a form element.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen



font-family fontFamily Bezeichner des Font-Familie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
fontWeight Sets or retrieves the numeric weight of the font of the object.
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontextes des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for an element.
max-width maxWidth Sets or retrieves the maximum width for an element.
min-height minHeight Sets or retrieves the minimum height for an element.
min-width minWidth Sets or retrieves the minimum width for an element.
overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-autospace textAutospace Autabstandsbehandlung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus
text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.17. Input checkbox

Beispiel für .checked

```
<HTML>  
<HEAD>  
</HEAD>  
<BODY>  
<INPUT type="checkbox" ID="XI1a" CHECKED onmousedown="OnMouseDown(0);">A<BR>  
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<INPUT type="checkbox" ID="XI1b">B<BR>  
<INPUT type="checkbox" ID="XI2" CHECKED>C<BR>  
<INPUT type="checkbox" ID="XI3" CHECKED>D<BR>  
<INPUT type="checkbox" ID="XI4a" CHECKED">E<BR>
```




```

<input type="checkbox" ID="XI4b">F<BR>
<input type="checkbox" ID="XI4c">G<BR>
<input type="checkbox" ID="XI4d">H<BR>
<input type="checkbox" ID="XI4e">I<BR>
<SCRIPT>
var X_Feld=new Array();
var X_Feld1=new Array();
X_Feld[0]=true;
X_Feld1[0]=XI1a;

```

```

function OnMouseDown(X00)
{ X_Feld[X00]=!X_Feld[X00]; // checked-Flag switchen
  X_Feld1[X00].checked=X_Feld[X00];
}
</SCRIPT>
</BODY>
</HTML>

```

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CHECKED checked Sets or retrieves the state of the check box or radio button.

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist

DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen

defaultChecked Sets or retrieves the state of the check box or radio button.

defaultValue Sets or retrieves the initial contents of the object.

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

form Retrieves a reference to the form that the object is embedded in.

HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

indeterminate Sets or retrieves whether the user has changed the status of a check box.

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Retrieves the document object associated with the node.

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Retrieves a value that indicates the current state of the object.

recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen

scrollWidth Scrollbreite vom Objekt ermitteln

SIZE size Sets or retrieves the size of the control.

sourceIndex Index des Objektes in der Collection document.all ermitteln



status Sets or retrieves the value indicating whether the control is selected.
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type Retrieves or initially sets the type of input control represented by the object.
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALUE value Sets or retrieves the default or selected value of the control.
 WIDTH width Sets or retrieves the calculated width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren



componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 select Highlights the input area of a form element.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen



direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 fontWeight Sets or retrieves the numeric weight of the font of the object.
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 max-height maxHeight Sets or retrieves the maximum height for an element.
 max-width maxWidth Sets or retrieves the maximum width for an element.
 min-height minHeight Sets or retrieves the minimum height for an element.
 min-width minWidth Sets or retrieves the minimum width for an element.
 overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
 overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
 overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.18. Input file

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters



clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

defaultValue Sets or retrieves the initial contents of the object.

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

form Retrieves a reference to the form that the object is embedded in.

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Retrieves the document object associated with the node.

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Retrieves a value that indicates the current state of the object.

scopeName Namensbereich des Objektes ermitteln

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen

scrollWidth Scrollbreite vom Objekt ermitteln

SIZE size Sets or retrieves the size of the control.

sourceIndex Index des Objektes in der Collection document.all ermitteln

STYLE style inline-Style des Objektes setzen

TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

TITLE title Tooltipp (Textblase bei mousevoer) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)

TYPE type Retrieves or initially sets the type of input control represented by the object.

uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen

value Retrieves the file name of the input object after the text is set by user input.

WIDTH width Sets or retrieves the calculated width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird

onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird

onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird

onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält

onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird

onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)

onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert



ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
onfocusin Fires for an element just prior to setting focus on that element.
onfocusout Fires for the current element with focus immediately after moving focus to another element.
onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
onmouseover Event erzeugt, sobald Maus das Objekt betritt
onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
onmousewheel Fires when the wheel button is rotated.
onmove Event erzeugt, wenn Objekt bewegt wird
onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
contains Ermitteln ob Objekt andere Objekte enthält
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
dragDrop Drag-Event für das Objekt erzeugen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
getAdjacentText Benachbarten Textstring liefern
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
getExpression Ausdruck für eine Eigenschaft liefern
hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
insertAdjacentElement Nachbar-Objekt einfügen
insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
insertAdjacentText Text als Nachbar-Text einfügen
insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
releaseCapture Maus-Event-Überwachung zum Objekt entfernen
removeAttribute Attribute aus einem Objekt entfernen
removeAttributeNode Attribut aus Objekt entfernen
removeBehavior Behavior aus einem Objekt entfernen
removeChild Kindknoten aus einem Objekt entfernen
removeExpression Ausdruck aus einer Eigenschaft entfernen
removeNode Objektknoten aus Dokument-Hierarchie entfernen
replaceAdjacentText Benachbarten Text ersetzen
replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
replaceNode Knoten durch anderen Knoten ersetzen
scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
select Highlights the input area of a form element.
setActive Objekt aktiv setzen, aber nicht Focus geben
setAttribute Wert eines Attributes setzen
setAttributeNode Attribut-Knoten in Objekt einfügen
setCapture Maus-Event-Überwachung zum Objekt starten



setExpression Ausdruck setzen

swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.

:hover :hover Sets the style of an element when the user hovers the mouse pointer over it.

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen

border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen

border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen

border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen

border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen

border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen

border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen

border-left borderLeft Linker Rahmen

border-left-color borderLeftColor Linker Rahmen

border-left-style borderLeftStyle Linker Rahmen

border-left-width borderLeftWidth Linker Rahmen

border-right borderRight Rechter Rahmen

border-right-color borderRightColor Rechter Rahmen

border-right-style borderRightStyle Rechter Rahmen

border-right-width borderRightWidth Rechter Rahmen

border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen

border-top borderTop Oberer Rahmen

border-top-color borderTopColor Oberer Rahmen

border-top-style borderTopStyle Oberer Rahmen

border-top-width borderTopWidth Oberer Rahmen

border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen

bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie

clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf

clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen

color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)

cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen

direction direction Leserichtung ermitteln bzw. setzen

display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)

filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen

font font font anhand Fonteneigenschaftenliste ermitteln bzw. setzen

font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen

font-size fontSize Höhe des Fonts ermitteln bzw. setzen

font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen

font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)

fontWeight Sets or retrieves the numeric weight of the font of the object.

font-weight fontWeight Font-Wichtung ermitteln bzw. setzen

height height Objekthöhe ermitteln bzw. setzen

layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen

layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen

left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)

letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen

line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen

margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen

margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen

margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen

margin-right marginRight Dicke von right-margin ermitteln bzw. setzen

margin-top marginTop Höhe von top-margin ermitteln bzw. setzen

max-height maxHeight Sets or retrieves the maximum height for an element.

max-width maxWidth Sets or retrieves the maximum width for an element.

min-height minHeight Sets or retrieves the minimum height for an element.

min-width minWidth Sets or retrieves the minimum width for an element.

overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf

overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf

overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf

padding padding Padding ermitteln bzw. setzen

padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen

padding-left paddingLeft Padding-left ermitteln bzw. setzen

padding-right paddingRight Padding-right ermitteln bzw. setzen

padding-top paddingTop Padding-Top ermitteln bzw. ersetzen

pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen



pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.19. Input radio

Beispiel für .checked

```

<HTML>
<HEAD>
</HEAD>
<BODY>
<INPUT type="radio" name="radio1" ID="XI1a" CHECKED onmousedown="OnMouseDown(0);">A <BR>
<INPUT type="radio" name="radio2" ID="XI1b" CHECKED>B<BR>
<INPUT type="radio" name="radio3" ID="XI2" CHECKED>C<BR>
<INPUT type="radio" name="radio4" ID="XI3" CHECKED>D<BR>
<INPUT type="radio" name="radio5" ID="XI4a" CHECKED>E<BR>
<INPUT type="radio" name="radio6" ID="XI4b" CHECKED>F<BR>
<INPUT type="radio" name="radio7" ID="XI4c" CHECKED>G <BR>
<INPUT type="radio" name="radio8" ID="XI4d" CHECKED>H <BR>
<INPUT type="radio" name="radio9" ID="XI4e" CHECKED>I <BR>
</SCRIPT>

```

// NAME="radiox" ist Gruppenname von Radio-Inputs, von denen zu jedem Zeitpunkt nur genau 1 markiert sein kann

```

var X_Feld=new Array();
var X_Feld1=new Array();
X_Feld[0]=true;
X_Feld1[0]=XI1a;

```

```

function OnMouseDown(X00) // onclick auf setzt immer checked auf true auch wenn schon Punkt da ist
//                           dazu muss nicht einmal ein Handler eingebunden sein, da Standard-Onclick aktiv
// onmousedown verändert zwar nicht selbst checked, impliziert onclick
// welche Reihenfolge die Eventverarbeitung hat, entscheidet der Browser.
//                           Damit gilt: Klick per Maus kann nicht zum switchen von checked genommen
werden.

```

```

//                           checked wird regelmäßig automatisch true sein.
{X_Feld[X00]!=X_Feld[X00]; // checked-Flag switchen
X_Feld1[X00].checked=X_Feld[X00];
// Es ist trotzdem wegen automatischen onclick checked auf true
// Wird allerdings alert() eingefügt, dann ist checked korrekt.
}

```

```

</SCRIPT>

```

```

</BODY>

```

```

</HTML>

```

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen



canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CHECKED checked Sets or retrieves the state of the check box or radio button.
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 defaultChecked Sets or retrieves the state of the check box or radio button.
 defaultValue Sets or retrieves the initial contents of the object.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 form Retrieves a reference to the form that the object is embedded in.
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontext um das Objekt ermitteln bzw. setzen
 outerText Text-Kontext um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 SIZE size Sets or retrieves the size of the control.
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 status Sets or retrieves the value indicating whether the control is selected.
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type Retrieves or initially sets the type of input control represented by the object.
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALUE value Sets or retrieves the default or selected value of the control.
 WIDTH width Sets or retrieves the calculated width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten



onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)

onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert

ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird

onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde

onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet

onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren

onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird

onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird

onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird

onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert

onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt

onmouseout Event erzeugt, sobald die Maus das Objekt verlässt

onmouseover Event erzeugt, sobald Maus das Objekt betritt

onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt

onmousewheel Fires when the wheel button is rotated.

onmove Event erzeugt, wenn Objekt bewegt wird

onmoveend Event erzeugt, wenn Bewegung eines Objektes endet

onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt

onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden

onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet

onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt

onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Referenz auf Attribut-Objekt nach Bezeichner der Eigenschaft liefern

getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

getExpression Ausdruck für eine Eigenschaft liefern

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

insertAdjacentHTML HTML-Code als Nachbar-Code einfügen

insertAdjacentText Text als Nachbar-Text einfügen

insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten

releaseCapture Maus-Event-Überwachung zum Objekt entfernen



removeAttribute Attribut aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 select Highlights the input area of a form element.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 fontWeight Sets or retrieves the numeric weight of the font of the object.
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen



margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for an element.
max-width maxWidth Sets or retrieves the maximum width for an element.
min-height minHeight Sets or retrieves the minimum height for an element.
min-width minWidth Sets or retrieves the minimum width for an element.
overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus
text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.20. Input text

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
AUTOCOMPLETE autoComplete Sets or retrieves the status of AutoComplete for the object.
canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
CLASS className Klassenname des Objektes ermitteln bzw. setzen
clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
defaultValue Sets or retrieves the initial contents of the object.
DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.
firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
form Retrieves a reference to the form that the object is embedded in.
HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
LANG lang zu benutzende Sprache ermitteln bzw. setzen



LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 MAXLENGTH maxLength Sets or retrieves the maximum number of characters that the user can enter in a text control.
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 READONLY readOnly Sets or retrieves the value indicated whether the content of the object is read-only.
 readyState Retrieves a value that indicates the current state of the object.
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 SIZE size Sets or retrieves the size of the control.
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type Retrieves or initially sets the type of input control represented by the object.
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALUE value Sets or retrieves the displayed value for the control object. This value is returned to the server when the control object is submitted.
 VCARD_NAME vcard_name Sets or retrieves the vCard value of the object to use for the AutoComplete box.
 WIDTH width Sets or retrieves the calculated width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onchange Fires when the contents of the object or selection have changed.
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschneitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.



onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onmouseover Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselect Event erzeugt wenn die Selektion des Objektes geändert wird
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 createTextRange TextRange-Objekt erzeugen
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 select Highlights the input area of a form element.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles



:first-child :first-child Applies one or more styles to any element that is the first child of its parent.

:hover :hover Sets the style of an element when the user hovers the mouse pointer over it.

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen

border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen

border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen

border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen

border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen

border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen

border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen

border-left borderLeft Linker Rahmen

border-left-color borderLeftColor Linker Rahmen

border-left-style borderLeftStyle Linker Rahmen

border-left-width borderLeftWidth Linker Rahmen

border-right borderRight Rechter Rahmen

border-right-color borderRightColor Rechter Rahmen

border-right-style borderRightStyle Rechter Rahmen

border-right-width borderRightWidth Rechter Rahmen

border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen

border-top borderTop Oberer Rahmen

border-top-color borderTopColor Oberer Rahmen

border-top-style borderTopStyle Oberer Rahmen

border-top-width borderTopWidth Oberer Rahmen

border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen

bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie

clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf

clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen

color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)

cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen

direction direction Leserichtung ermitteln bzw. setzen

display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)

filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen

font font anhand Fonteigenschaftenliste ermitteln bzw. setzen

font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen

font-size fontSize Höhe des Fonts ermitteln bzw. setzen

font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen

font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)

fontWeight Sets or retrieves the numeric weight of the font of the object.

font-weight fontWeight Font-Wichtung ermitteln bzw. setzen

hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat

height height Objekthöhe ermitteln bzw. setzen

ime-mode imeMode Sets or retrieves the state of an Input Method Editor (IME).

layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen

layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen

left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)

letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen

line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen

margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen

margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen

margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen

margin-right marginRight Dicke von right-margin ermitteln bzw. setzen

margin-top marginTop Höhe von top-margin ermitteln bzw. setzen

max-height maxHeight Sets or retrieves the maximum height for an element.

max-width maxWidth Sets or retrieves the maximum width for an element.

min-height minHeight Sets or retrieves the minimum height for an element.

min-width minWidth Sets or retrieves the minimum width for an element.

overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf

overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf

overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf

padding padding Padding ermitteln bzw. setzen

padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen

padding-left paddingLeft Padding-left ermitteln bzw. setzen

padding-right paddingRight Padding-right ermitteln bzw. setzen

padding-top paddingTop Padding-Top ermitteln bzw. ersetzen

pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen

pixelHeight Höhe des Objektes ermitteln bzw. setzen



pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, an der Fluss erfolgt
 text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.21. LINK

kann nur innerhalb HEAD hinterlegt werden

Beispiel: <LINK REL=stylesheet HREF="styles.css" type="text/css">

für Besonderheiten siehe auch A-Objekt

Attribute

canHaveHTML Sets or retrieves the value indicating whether the object can contain rich HTML markup.
 charset Zeichensatz zum Encodieren des Objektes ermitteln bzw. setzen
 DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 HREF href Ankerzielpunkt ermitteln bzw. setzen
 HREFLANG hreflang Sets or retrieves the language code of the object.
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltyps des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 REL rel Beziehungen zwischen Objekt und Linkziel ermitteln bzw. setzen
 REV rev Beziehungen zwischen Objekt und Linkziel ermitteln bzw. setzen
 scopeName Namensbereich des Objektes ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TARGET target Zielfenster bzw. Ziel-Frame ermitteln bzw. setzen
 TYPE type MIME-Typ des Objektes ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

Events

onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden



addBehavior Dem Objekt ein Behavior hinzufügen
applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
contains Ermitteln ob Objekt andere Objekte enthält
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
getAdjacentText Benachbarten Textstring liefern
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
insertAdjacentElement Nachbar-Objekt einfügen
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
removeAttribute Attribute aus einem Objekt entfernen
removeAttributeNode Attribut aus Objekt entfernen
removeBehavior Behavior aus einem Objekt entfernen
replaceAdjacentText Benachbarten Text ersetzen
setAttribute Wert eines Attributes setzen
setAttributeNode Attribut-Knoten in Objekt einfügen
swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
MEDIA media Sets or retrieves the media type.
pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen

4.3.2.2.12.9.22. location

enthält Informationen zu im Browser angewählten Urls

Attribute

hash Sets or retrieves the subsection of the href property that follows the number sign (#).
host Sets or retrieves the hostname and port number of the location or URL.
hostname Sets or retrieves the host name part of the location or URL.
href Sets or retrieves the entire URL as a string.
pathname Dateiname aus dem Pfad des Objektes ermitteln bzw. setzen
port Sets or retrieves the port number associated with a URL.
protocol Sets or retrieves the protocol portion of a URL.
search Sets or retrieves the substring of the href property that follows the question mark.

Events

die von Fenster und vom Dokument, das per Url angewählt wird

Methoden

assign Loads a new HTML document.
reload Reloads the current page.
replace Replaces the current document by loading another document at the specified URL.

Styles

die von Fenster und vom Dokument, das per Url angewählt wird

4.3.2.2.12.9.23. MARQUEE

Laufschrift

Beispiel:

```

<MARQUEE DIRECTION=RIGHT BEHAVIOR=SCROLL SCROLLAMOUNT=10 SCROLLDELAY=200>
Das ist eine Laufschrift.
</MARQUEE>
  
```

Beispiel:



```

<MARQUEE id=m1 direction=right style="border-width:2px;border-style:solid;"
width=200 height=200>right</MARQUEE>

<BR>
<BUTTON onclick="alert('scrollLeft: ' + m1.scrollLeft + ' scrollRight: ' + m1.scrollTop)"> Scrolldaten anzeigen</BUTTON>
<BUTTON onclick="m1.stop();m1.scrollLeft = 190;">Stop & Set scrollLeft=190</BUTTON>
<BUTTON onclick="m1.start();">Start</BUTTON>

```

Standardbreite ist die Breite des Containers, in dem MARQUEE liegt (Eltern-Container)

In TD einer Tabelle:

wenn innerhalb TD so Breite vom MARQUEE muss angegeben werden, wenn TD ohne Breitenangabe (da sonst 1 Pixel als Breite verwendet wird)

Senkrechte Laufschrift: .scrollLeft auf 0 setzen

Horizontale Laufschrift: .scrollTop auf 0 setzen

.scrollLeft und .scrollTop nur belegbar, wenn Marquee nicht animiert (während Animation nur lesbar)

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

BEHAVIOR behavior Sets or retrieves how the text scrolls in the marquee.

BGCOLOR bgColor Diese Eigenschaft ist nicht mehr verwendbar !

canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen

DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist

DATAFORMATAS dataFormatAs Art der Renders von Daten im Objekt ermitteln bzw. setzen

DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

DIRECTION direction Sets or retrieves the direction in which the text should scroll.

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

Height height Objekthöhe ermitteln bzw. setzen

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

HSPACE hspace Sets or retrieves the horizontal margin for the object.

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

LOOP loop Sets or retrieves the number of times a marquee will play.

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Retrieves the document object associated with the node.

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Status des Objektes ermitteln

recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

SCROLLAMOUNT scrollAmount Sets or retrieves the number of pixels the text scrolls between each subsequent drawing of the marquee.

SCROLLDELAY scrollDelay Sets or retrieves the speed of the marquee scroll.

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen



sourceIndex Index des Objektes in der Collection document.all ermitteln

STYLE style inline-Style des Objektes setzen

TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)

TRUESPEED trueSpeed Sets or retrieves whether the position of the marquee is calculated using the scrollDelay and scrollAmount properties and the actual time elapsed from the last clock tick.

uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen

VSPACE vspace Sets or retrieves the vertical margin for the object.

WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird

onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten

onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird

onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird

onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält

onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird

onbeforeupdate Event erzeugt direkt vor dem Update der Daten

onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)

onbounce Fires when the behavior property of the marquee object is set to "alternate" and the contents of the marquee reach one side of the window.

onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschneitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert

ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird

onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde

onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet

onfinish Fires when marquee looping is complete.

onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren

onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird

onkeydown Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird

onkeydown Event erzeugt, sobald eine gedrückte Taste losgelassen wird

onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert

onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt

onmouseout Event erzeugt, sobald die Maus das Objekt verlässt

onmouseover Event erzeugt, sobald Maus das Objekt betritt

onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt

onmousewheel Fires when the wheel button is rotated.

onmove Event erzeugt, wenn Objekt bewegt wird

onmoveend Event erzeugt, wenn Bewegung eines Objektes endet

onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt

onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden

onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert

onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet

onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt

onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)

onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

onstart Fires at the beginning of every loop of the marquee object.

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind



applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 start Starts scrolling the marquee.
 stop Stops the marquee from scrolling.
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :first-letter :first-letter Layout der ersten Buchstaben im Objekt festlegen
 :first-line :first-line Layout der ersten Zeile im Objekt festlegen
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color background-color Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen



border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
fontWeight Sets or retrieves the numeric weight of the font of the object.
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for an element.
min-height minHeight Sets or retrieves the minimum height for an element.
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus
text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)



unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.24. navigator

Informationen zum Browser, siehe auch Objekt clientInformation und userProfile

Attribute

appName Codename des Browsers
 appMinorVersion Unterversion des Browsers
 appName Anwendungsname des Browsers
 appVersion Betriebssystemplattform und Version des Browser
 browserLanguage Sprache im Browser als Anwendung
 cookieEnabled Status ob Cookies dauerhaft ermöglicht werden oder nunr temporär
 cpuClass Bezeichner der Prozessor-Klasse (CPU class)
 onLine Status ob System global on- oder offline
 platform Betriebssystemplattform des Users
 systemLanguage Standardsprache des Betriebssystems
 userAgent HTTP-User-Agent
 userLanguage Sprache im System

Events

keine

Methoden

javaEnabled Ermitteln ob Java verfügbar ist
 taintEnabled Ermitteln ob data tainting möglich ist

Styles

keine

Objekte

userProfile Objekt zur Verwaltung (auch Speichern) von Userdaten im Datenbereich des Internet Explorers

Collectionen

plugins Collection aller eingebetteten EMBED-Objekte im Dokument
 Diese Collection existiert nur zum Zweck der Komaptibilität mit Nicht-IE-Browsern, da der IE
 in der Regel EMBED nicht nutzt: Anstelle von Plugins werden z.B. Actvie-X verwendet.
 Stellt ein Plugin-Anbieter kein Active-X-Control bereits, so muss das Plugin verwendet werden.
 Syntax

```
[ oColl = ] object.plugins
[ oObject = ] object.plugins(iIndex)

oColl Array that is empty.
oObject Reference to an individual item in the array of elements contained by the object.
iIndex Required. Integer that specifies the zero-based index of the item to be returned.
```

Attribut: length Sets or retrieves the number of objects in a collection.

Methoden

```
item()           Retrieves an object from the all collection or various other collections.
namedItem()      Retrieves an object or a collection from the specified collection.
tags()           Retrieves a collection of objects that have the specified HTML tag name.
```

4.3.2.2.12.9.25. OBJECT

OBJECT hat kein .innerHTML

Beispiel für Erzeugung eines Test-Objektes

```
function test(X00)
// X00 CLASSID-String mit 'CLSID:' z.B. 'CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D'
// Es erfolgt keine Syntaxprüfung !
// wenn Leerkette so kein Testobjekt erzeugt
// liefert true, wenn Testobjekt erzeugbar war
// false, wenn document.body nicht vorhanden
// Parameterfehler
// Testobjekt nicht erzeugbar
// Das temporäre Testobjekt ist NICHT mehr verfügbar mit Ende der Funktion !
{var X01;
var X02;
var X03=false;
var X04="";
// +++++ document.body prüfen
X03=(document.body!=null);
// +++++ Parameter prüfen
if(X03){X03=(X00!=null);if(X03){X03=(X00!="");}}
```




```
// +++++ Objekt dynamisch erzeugen
if(X03)
{ X01=document.createElement('OBJECT');           // kein TAG-Begrenzer und NICHT CLASSID belegen !
  X03=(X01!=null);
}
if(X03)
{ X02=document.body.appendChild(X01);
  X03=(X02!=null);
}
if(X03){ X03=(X01==X02);}                          // beide Zeiger identisch ?
if(X03)
{ // +++++ Objekt unsichtbar machen
  X02.style.visibility='hidden';
  // +++++ CLASSID per try-catch zuweisen
  try{ X02.classid=X00;} catch(e){ X03=false;}        // e ist Platzhalter für Fehlercode, der aber nicht ausgewertet wird
  // +++++ classid-Belegung prüfen
  if(X03){ X03=(X02.classid!="");}                  // und prüfen anstelle Fehlercode-Auswertung
  // +++++ Objekt aus DOM entfernen
  X01=document.body.removeChild(X02);               // Testobjekt aus DOM entfernen
}
return X03;
}
```

Eigenschaft .object

Zeiger auf das Objekt im OBJECT

Beispiel für OBJECT-Fehler in HTML

```
<OBJECT CLASSID="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">
<SPAN STYLE="color:red">Objekt wurde nicht geladen ! </SPAN>
</OBJECT>
```

Achtung: Auch wenn der SPAN nicht aktiviert wird, dann heisst das nicht, dass das Objekt wirklich funktioniert. Es können z.B. Laufzeitbibliotheken zum Objekt nicht mehr verfügbar sein, so dass deren Aufruf einen Runtime-Fehler bringen kann. Es muss also zusätzlich auf Funktionstüchtigkeit der Runtime-Bibliotheken geprüft werden - z.B. mit JScript.

JHTML-Datei als Scriptlet nachladen:

```
<object type="text/x-scriptlet" width=100% height="100" data="1.htm"></object>
```

Beispiele für Active-X-Control des Mediaplayers zum Abspielen eines Videos oder Sounds:

Achtung: CLASSID wird inzwischen nicht mehr sicher unterstützt (siehe unten Microsoft ändert die Active-X-Eigenschaften

```
<BODY>
<OBJECT ID="ActiveMovie1"
  WIDTH=356
  HEIGHT=328
  CLASSID="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A"
>
  <PARAM NAME="MovieWindowWidth" VALUE="352">
  <PARAM NAME="MovieWindowHeight" VALUE="247">
  <PARAM NAME="FileName" VALUE="0.wmv">
</OBJECT>
```

Das Active-X-Control umfasst folgende PARAM

```
<PARAM NAME="Appearance" VALUE="0">
<PARAM NAME="AutoStart" VALUE="0">
<PARAM NAME="AllowChangeDisplayMode" VALUE="-1">
<PARAM NAME="AllowHideDisplay" VALUE="0">
<PARAM NAME="AllowHideControls" VALUE="-1">
<PARAM NAME="AutoRewind" VALUE="-1">
<PARAM NAME="Balance" VALUE="0">
<PARAM NAME="CurrentPosition" VALUE="0">
<PARAM NAME="DisplayBackColor" VALUE="0">
<PARAM NAME="DisplayForeColor" VALUE="16777215">
<PARAM NAME="DisplayMode" VALUE="0">
<PARAM NAME="Enabled" VALUE="-1">
<PARAM NAME="EnableContextMenu" VALUE="-1">
<PARAM NAME="EnablePositionControls" VALUE="-1">
<PARAM NAME="EnableSelectionControls" VALUE="0">
```



```

<PARAM NAME="EnableTracker" VALUE="-1">
<PARAM NAME="Filename" VALUE="">
<PARAM NAME="FullScreenMode" VALUE="0">
<PARAM NAME="MovieWindowSize" VALUE="0">
<PARAM NAME="PlayCount" VALUE="1">
<PARAM NAME="Rate" VALUE="1">
<PARAM NAME="SelectionStart" VALUE="-1">
<PARAM NAME="SelectionEnd" VALUE="-1">
<PARAM NAME="ShowControls" VALUE="-1">
<PARAM NAME="ShowDisplay" VALUE="-1">
<PARAM NAME="ShowPositionControls" VALUE="0">
<PARAM NAME="ShowTracker" VALUE="-1">
<PARAM NAME="Volume" VALUE="-600">

```

VALUE-Angaben sind nur Beispiele

Bezeichner der Eigenschaften für JScript sind IDENTISCH mit den Werten aus NAME-Attribut von PARAM !
Gross-Kleinschreibung beachten !

User	AllowChangeDisplayMode	Änderung des Anzeigemodus Zeitanzeige bzw. Framesanzeige durch User boolean true oder false
	AllowHideControls	Änderung des Anzeigemodus der Steuerungstasten während Wiedergabe durch User boolean true oder false false (0) für nicht erlauben
	AllowHideDisplay	Änderung des Anzeigemodus der Wiedergabe-Anzeige während Wiedergabe durch boolean true oder false false (0) für nicht erlauben
	AutoRewind	Automatisches Rückspulen nach Stop boolean true oder false false (0) für nicht rückspulen
	AutoStart	Autostart der Wiedergabe ein oder aus boolean true oder false false (0) für nicht automatische Wiedergabe: siehe Aktionen zum Player
	Balance	numerisch 0 für zentriert default -1000 ganz links +1000 ganz rechts
	CurrentPosition	unklar
	DisplayBackColor	Hintergrundfarbe der Control-Elemente numerischer Farbwert, kein Hexastring 0 Default für Black 16777215 für Weiss
	DisplayForeColor	Hintergrundfarbe der Control-Elemente numerischer Farbwert, kein Hexastring 0 Default für Black 16777215 für Weiss
	DisplayMode	Anzeigeart der Position 0 für Time 1 für Frames
	Enabled	Control-Anzeige ein, aus boolean true oder false false (0) für nicht erlauben
	EnableContextMenu	Rechte-Maus-Kontextmenü ein, aus boolean true oder false false (0) für nicht erlauben
	EnablePositionControls	Positionierungs-Button im Control-Panel ein, aus boolean true oder false false (0) für nicht erlauben
	EnableSelectionControls	Selektions-Button im Control-Panel ein, aus boolean true oder false



	false (0) für nicht erlauben
EnableTracker	Tracker-Balken im Control-Panel ein, aus boolean true oder false false (0) für nicht erlauben
FileName	Mediendatei mit Pfad oder Url
FullScreenMode	Vollbildschirm ein, aus boolean true oder false false (0) für kein Vollbildmodus
MovieWindowSize	unklar, Grösse des Anzeigefensters
PlayCount	Anzahl der Wiedergaben der Mediendatei ab 1
Rate	Wiedergabefaktor numerisch 1 für 1-fache Geschwindigkeit 10 für 20-fache Geschwindigkeit
SelectionEnd	Bis-Position, bis zu der wiedergegeben werden soll in Sekunden bezüglich Anfang der Mediendatei (0)
SelectionStart	Von-Position, bis zu der wiedergegeben werden soll in Sekunden bezüglich Anfang der Mediendatei (0)
ShowControls	Start Stop buttons and tracker controls sichtbar ein, aus boolean true oder false false (0) für aus
ShowDisplay	Status-Anzeige-Panel sichtbar ein, aus boolean true oder false false (0) für aus
ShowPositionControls	Previous, Rewind, Forward and Next buttons sichtbar ein, aus boolean true oder false false (0) für aus
ShowSelectionControls	Selektions-Button im Control-Panel sichtbar ein, aus boolean true oder false false (0) für aus
ShowTracker	Trackerbar im Controlpanel sichtbar ein, aus: nur lesen boolean true oder false false (0) für aus
uiMode	Sichtbarkeit anstelle STYLE-visible "invisible" für unsichtbar
Volume	Laustärke als Basis dient die aktuelle Windows-Lautstärke integer 0 für maximale, also aktuelle Windows-Lautstärke (höher geht nicht !!!) -10000 für total still

Den ganzen Mediaplayer unsichtbar machen: per STYLE-Attribut im OBJECT-Tag

document.ID-Bezeichner.readyState (Mediaplayer 6.4, funktioniert aber auch mit höheren Versionen)

.readyState hat folgende Werte

0	The FileName property has not been initialized.
1	Windows Media Player control is asynchronously loading a file.
3	Windows Media Player control loaded a file, and downloaded enough data to play the file, but has not yet received all data.
4	All data has been downloaded.

Methoden:

.IsSoundCardEnabled()

liefert boolean



true Soundkarte erkannt
false keine Soundkarte erkannt

.Run(); Erzeugt und startet Instanz des Mediaplayers und ermöglicht danach weitere Aktionen wie .Play etc.

.Mute ohne ()

lesen, schreiben
true Player volume is muted.
false (Default) Player volume is not muted.

.Play ohne ()

Wiedergabe starten

.AutoStart ohne ()

true (Default) Clip automatically starts.
false Disables automatic start.

.Stop ohne ()

.Pause ohne ()

Audio wiedergeben

```
<HEAD>
<SCRIPT>
var SoundStarten_TimeoutID=0;

function SoundStarten()
{
  // auf Soundkarte prüfen
  if (ID_PlayerObjekt.IsSoundCardEnabled())
  {
    // Soundkarte vorhanden
    // Musik geladen und abspielbar ?
    if (ID_PlayerObjekt.readyState == 4)
    {
      // geladen, also Playerinstanz starten
      ID_PlayerObjekt.Run();
      // Musik wiedergeben
      ID_PlayerObjekt.Play;
      alert('Musik ist gestartet !');
    }
    else{ SoundStarten_TimeoutID=window.setTimeout('SoundStarten()', 100);}
  }
  else{ alert('Keine Soundkarte gefunden !');}
}
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_PlayerObjekt" CLASSID="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A">
  <PARAM NAME="FileName" VALUE="a0.mp3">
  <PARAM NAME="AutoStart" VALUE="0">      <!-- kein Autoplay -->
</OBJECT>
<SCRIPT>
if (ID_PlayerObjekt != null){SoundStarten();}
</SCRIPT>
</BODY>
</HTML>
```

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.



Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899
Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit Kompression genutzt wird z.B. bei
onclick-Handler auf IMG
klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren

oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemerkt aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer

anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen

Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.



Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
zwischen Register in einem IE-Fenster
zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
.setActive() ist Teilmenge von .focus(): nur das aktiv setzen
funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht



es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement {333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.



Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Umeine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5



IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}
```

```
var X85=new Array();var X86=new Array();
```

```
X85[0]=window.open(...);
```



```
var X87='parent.Y_unload(0);'; X86[0]=new Function(",X87);
```

```
X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet,
so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr
wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">  
unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild
```

```
<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12) ">  
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12) ">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standardgemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,

innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen

ALIGN align Sets or retrieves how the object is aligned with adjacent text.

ALT alt Sets or retrieves a text alternative to the graphic.

altHTML Sets the optional alternative HTML script to execute if the object fails to load.

ARCHIVE archive Sets or retrieves a character string that can be used to implement your own archive functionality for the object.

BaseHref Retrieves a string of the URL where the object tag can be found. This is often the href of the document that the object is in, or the value set by a base element.

BORDER border Sets or retrieves the width of the border to draw around the object.

canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CLASSID classid Sets or retrieves the class identifier for the object.

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

CODE code Sets or retrieves the URL of the file containing the compiled Java class.

CODEBASE codeBase Sets or retrieves the URL of the component.

CODETYPE codeType Sets or retrieves the Internet media type for the code associated with the object.

DATA data Sets or retrieves the URL that references the data of the object.

DECLARE declare Sets or retrieves a character string that can be used to implement your own declare functionality for the object.

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

form Retrieves a reference to the form that the object is embedded in.

Height height Objekthöhe ermitteln bzw. setzen

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

HSPACE hspace Sets or retrieves the horizontal margin for the object.

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

object Retrieves the contained object.

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen



parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves the current state of the object.
 recordset Sets or retrieves from a data source object a reference to the default record set.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STANDBY standby Sets or retrieves a character string that can be used to implement your own standby functionality for the object.
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 TYPE type MIME-Typ des Objektes ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 USEMAP useMap Sets or retrieves the URL, often with a bookmark extension (#name), to use as a client-side image map.
 VSPACE vspace Sets or retrieves the vertical margin for the object.
 WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 oncellchange Fires when data changes in the data provider.
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 ondataavailable Fires periodically as data arrives from data source objects that asynchronously transmit their data.
 ondatasetchanged Fires when the data set exposed by a data source object changes.
 ondatasetcomplete Fires to indicate that all data is available from the data source object.
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerror Fires when an error occurs during object loading.
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeydown Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onrowenter Fires to indicate that the current row has changed in the data source and new data values are available on the object.
 onrowexit Fires just before the data source control changes the current row in the object.
 onrowsdelete Fires when rows are about to be deleted from the recordset.
 onrowsinserted Fires just after new rows are inserted in the current recordset.
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten



detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getExpression Ausdruck für eine Eigenschaft liefern
 insertAdjacentElement Nachbar-Objekt einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 namedRecordset Retrieves the recordset object corresponding to the named data member from a data source object (DSO).
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links rechts oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen



padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 scrollbar-3dlight-color scrollbar3dLightColor Farbe der oberen und linken Kante von Scroobar und Scroll-Pfeile ermitteln bzw. setzen
 scrollbar-arrow-color scrollbarArrowColor Farbe der Scrol-Pfeile ermitteln bzw. setzen
 scrollbar-base-color scrollbarBaseColor Farbe der Hauptelement der Scrollbar, Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-darkshadow-color scrollbarDarkShadowColor Schattenfarbe der Scrollbar ermitteln bzw. setzen
 scrollbar-face-color scrollbarFaceColor Farbe der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-highlight-color scrollbarHighlightColor Farbe der oberen und linken Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-shadow-color scrollbarShadowColor Farbe der unteren und der rechten Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-track-color scrollbarTrackColor Farbe der Trackelemente der Scrollbar ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.26. OPTION

Attribute

BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 defaultSelected Sets or retrieves the status of the option.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 DISABLED disabled Sets or retrieves a value that you can use to implement your own disabled functionality for the object.
 END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 form Retrieves a reference to the form that the object is embedded in.
 hasMedia Status (booealan) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 index Sets or retrieves the ordinal position of an option in a list box.
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LABEL label Sets or retrieves a value that you can use to implement your own label functionality for the object.
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln



parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 SELECTED selected Sets or retrieves whether the option in the list box is the default item.
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
 systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der User-Einstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 text Sets or retrieves the text string specified by the option tag.
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 VALUE value Sets or retrieves the value which is returned to the server when the form control is submitted.

Events

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern- oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen



behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
direction direction Leserichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontext des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
width width Breite eines Objektes ermitteln bzw. setzen
word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.27. P**Attribute**

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
ALIGN align Ausrichtung des Objektes beim Rendern
ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
blockDirection Richtung des Block-Element-Flusses (links nach rechts oder umgekehrt) ermitteln bzw. setzen
canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
CLASS className Klassenname des Objektes ermitteln bzw. setzen
clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
DIR dir Leserichtung des Objektes ermitteln bzw. setzen
disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen
END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
hasMedia Status (boolean) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist
HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
LANG lang zu benutzende Sprache ermitteln bzw. setzen
LANGUAGE language Scriptsprache ermitteln bzw. setzen
lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
nodeName Name des Teiltypen des Knoten ermitteln
nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
nodeValue Wert eines Knoten ermitteln bzw. setzen
offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
outerHTML HTML-Kontext um das Objekt ermitteln bzw. setzen
outerText Text-Kontext um das Objekt ermitteln bzw. setzen
ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist



parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
 systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
 onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Event erzeugt, wenn Mausrad gedreht wird
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet



onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt

onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln

getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

getExpression Ausdruck für eine Eigenschaft liefern

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

insertAdjacentHTML HTML-Code als Nachbar-Code einfügen

insertAdjacentText Text als Nachbar-Text einfügen

insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten

releaseCapture Maus-Event-Überwachung zum Objekt entfernen

removeAttribute Attribute aus einem Objekt entfernen

removeAttributeNode Attribut-Objekt aus einem Objekt entfernen

removeBehavior Behavior aus einem Objekt entfernen

removeChild Kindknoten aus einem Objekt entfernen

removeExpression Ausdruck aus einer Eigenschaft entfernen

removeNode Objektknoten aus Dokument-Hierarchie entfernen

replaceAdjacentText Benachbarten Text ersetzen

replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen

replaceNode Knoten durch anderen Knoten ersetzen

scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)

setActive Objekt aktiv setzen, aber nicht Focus geben

setAttribute Wert eines Attributes setzen

setAttributeNode Attribut-Objekt-Knoten einfügen

setCapture Maus-Event-Überwachung zum Objekt starten

setExpression Ausdruck setzen

swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-letter :first-letter Layout der ersten Buchstaben im Objekt festlegen

:first-line :first-line Layout der ersten Zeile im Objekt festlegen

ACCELERATOR accelerator Tastaturkürzel setzen

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen

border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen

border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen

border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen

border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen

border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen

border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen

border-left borderLeft Linker Rahmen

border-left-color borderLeftColor Linker Rahmen

border-left-style borderLeftStyle Linker Rahmen

border-left-width borderLeftWidth Linker Rahmen

border-right borderRight Rechter Rahmen



border-right-color borderRightColor Rechter Rahmen
border-right-style borderRightStyle Rechter Rahmen
border-right-width borderRightWidth Rechter Rahmen
border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
direction direction Leserichtung ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-char layoutGridChar Gittergrösse für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
text-align-last textAlignLast Ausrichtung der letzten Text-Linie im Objekt ermitteln bzw. setzen
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus



text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
 text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-kashida-space textKashidaSpace Layout kashida ermitteln bzw. setzen
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können
 width width Breite eines Objektes ermitteln bzw. setzen
 word-break wordBreak Zeilenumbruch in wörtern bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.28. PARAM**Attribute**

DATAFLD Sets the field of a given data source for data binding.
 DATAFORMATAS Sets whether data supplied to the object should be rendered as text or HTML.
 DATASRC Sets the source of the data for data binding.
 NAME name Sets or retrieves the name of an input parameter for an element.
 TYPE type Sets or retrieves the content type of the resource designated by the value attribute.
 VALUE value Sets or retrieves the value of an input parameter for an element.
 VALUETYPE valueType Sets or retrieves the data type of the value attribute.

Events

keine

Methoden

keine

Styles

keine

4.3.2.2.12.9.29. popup**Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers**

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.



Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren

oder Popsups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer

anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

window.focus();

window.document.focus();

if(document.body!=null)

{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)

{document.body.focus();}

}

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

popupzeiger.show(...);

Hinweis: Der Populfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste



Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtsstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X-Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)



Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}
Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes
• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer



Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements



wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87='parent.Y_unload(0);'; X86[0]=new Function("X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet, so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild
```

```
<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12)">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12)">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').



Grund: Es wird standargemäß immer am Ende des BODY angefügt.
 Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,
 innerhalb
 dessen dann die neuen HTML-Elemente erzeugt werden.

Popup ist nicht identisch mit Fenster per `window.open()`

Ein Popop besitzt keine Eigenschaften wie ID
 kennt keine Variablen und Funktionen von Aufrufer des Popups
 schränkt Eigenschaften und Methoden eines im innerHTML liegendes HTML-Elementes ein
 Bsp: IMG als innerHTML
 per onmouseover-Handler des IMG kodiertes
`parentElement.hide();` bringt Scriptfehler
`parentNode.hide();` bringt Scriptfehler
`this.fireEvent('onclick');` funktioniert nicht
`alert();` funktioniert

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per `window.createPopup()`
 Popup per `window`-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.
 Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.
 Der Fehler tritt nicht auf, wenn ein Fenster per `window.open()` erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7
 Popupblocker ist im IE abgeschaltet
 ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 `window.popup` per `.show()` erzeugt.
 ein weiteres Fenster (Register) z.B. leere Seite (`about:blank`)
 beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per `.show()` erzeugt,
 bricht der Browser das Dokument mit `.show()` ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit `.show()`. Es wird folgende
 Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Popupfenster geblockt. Sie können den Popupblocker deaktivieren
 oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter
 Popupblocker einschalten
 weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster
 einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer
 anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen
 Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

`onfocus`
`onblur`
`onfocusin`
`onfocusout`

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
{document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupefehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von
 Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7



windows.focus() document.focus() und body.focus() funktionieren NICHT
zwischen Register in einem IE-Fenster
zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
.setActive() ist Teilmenge von .focus(): nur das aktiv setzen
funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

Beispiel für Erzeugen und Anzeigen eines Popup:

```
var Popup = window.createPopup();
var PopupBody = Popup.document.body;
PopupBody.innerHTML = "Display some <B>HTML</B> here.";
Popup.show(100, 100, 200, 50, document.body); // immer document.body
```

.innerHTML ist identisch mit .innerText. Per .innerText ist HTML-Code zuweisbar.

Hintergrundbild in einem Popup

Transparente GIF-Dateien werden nicht-zwingend transparent gerendert.
popupzeiger.document.body.style.backgroundColor='transparent'; // wird nicht gerendert
popupzeiger.document.body.style.background='transparent url(+ Datei + ') no-repeat fixed left top';
// Transparenz nicht gerendert

Animierte GIF-Dateien werden nicht zwingend animiert gerendert.

Variante 1: popupzeiger.document.body.style.backgroundImage='url(+ 'test.gif' + ')';
oder popupzeiger.document.body.style.background='transparent url(+ Datei + ') no-repeat fixed left top';
Belegung geht NUR per url(). Es geht keine Wertzuweisung anhand .scr.
Achtung: url() existiert nicht als Zeiger
var Kette=url() geht also nicht
ist interne Funktion

Variante 2: popupzeiger.document.body.innerHTML='';

anstelle von zeiger_auf_popup.document.body.style.backgroundImage
das entweder url(zeichenkette) oder 'none' haben darf,
dafür NICHT .src als Wert eines per new Image() erzeugten Objektes
ist auch zeiger_auf_popup.document.body.background
verwendbar, das mit .src eines per new Image() erzeugten Objektes belegt werden kann.

Ist das Popup in der Dimension kleiner als das Hintergrundbild, dann wird das Hintergrundbild (egal ob Gif oder JPG etc.) nicht
komplett angezeigt, sondern abgeschnitten.

Wird ein Eventhandler für

popupzeiger.document
oder popupzeiger.document.body
eingebunden, so ist popupzeiger.event nicht verfügbar, da event von window stammt (genauso wenig geht
window.popupzeiger.event)

Das Schliessen eines Popup durch den Userklick ausserhalb des Popup

löscht nicht die erzeugte Instanz des Popup
verändert .isOpen

In einem Popup kann Text markiert werden.

Attribute

document Retrieves the HTML document in a given popup window.
isOpen Retrieves a value indicating whether the popup window is open.
die von popup_zeiger.document
die von popup_zeiger.document.body

Events

die von popup_zeiger.document
die von popup_zeiger.document.body

Methoden

hide Closes the pop-up window.
show Displays the pop-up window on the screen.
siehe auch window.createPopup()
die von popup_zeiger.document
die von popup_zeiger.document.body

Styles

writing-mode writingMode Sets or retrieves the direction and flow of the content in the object.
die von popup_zeiger.document
die von popup_zeiger.document.body



4.3.2.2.12.9.30. screen**Attribute**

availHeight Retrieves the height of the working area of the system's screen, excluding the Microsoft® Windows® taskbar.

availWidth Retrieves the width of the working area of the system's screen, excluding the Windows taskbar.

bufferDepth Sets or retrieves the number of bits per pixel used for colors in the off-screen bitmap buffer.

colorDepth Retrieves the number of bits per pixel used for colors on the destination device or buffer.

deviceXDPI Retrieves the actual number of horizontal dots per inch (DPI) of the system's screen.

deviceYDPI Retrieves the actual number of vertical dots per inch (DPI) of the system's screen.

fontSmoothingEnabled Retrieves whether the user has enabled font smoothing in the Display control panel.

height Retrieves the vertical resolution of the screen.

logicalXDPI Retrieves the normal number of horizontal dots per inch (DPI) of the system's screen.

logicalYDPI Retrieves the normal number of vertical dots per inch (DPI) of the system's screen.

updateInterval Sets or retrieves the update interval for the screen.

width Retrieves the horizontal resolution of the screen.

Events

keine

Methoden

keine

Styles

keine

4.3.2.2.12.9.31. SCRIPT

Script dynamisch erzeugen am Beispiel VBScript

VBScript lässt sich NUR dynamisch erzeugen mit der eigenschaft .text

vbscript_objekt_zeiger.canHaveHTML liefert false: Damit ist .innerHTML NUR LESBAR

vbscript_objekt_zeiger.innerHTML='wert' wird NICHT ausgeführt !

vbscript_objekt_zeiger.innerText='wert'; wird ebenfalls NICHT ausgeführt

vbscript_objekt_zeiger.text wird ausgeführt

```
<html>
<body>
<script>
var X01;
var X02;
var X03=false;
var X04="";
var X05=0;
X04='SCRIPT';

function JSTest(X00,X01)
{alert('JSTest() ' + X00 + ' ' + X01);}

X01=document.createElement(X04);X03=(X01!=null);
if(X03){ X02=document.body.appendChild(X01);X03=(X02!=null);}
if(X03){X03=(X01==X02);}
if(X03){
alert('Tagname = ' + X01.tagName);
X01.language='VBScript';alert('LANGUAGE = ' + X01.language);
alert('.innerText vorhanden ' + (X01.innerText!=null));
alert('.innerText = ' + X01.innerText);
alert('Kann HTML zwischen Tags haben = ' + X01.canHaveHTML);
alert('.innerHTML vorhanden ' + (X01.innerHTML!=null));
alert('.innerHTML = ' + X01.innerHTML);
alert('.text vorhanden ' + (X01.text!=null));
alert('.text = ' + X01.text);
X01.text='sub VBTest() JSTest 0,1 end sub';
alert('.text = ' + X01.text);
VBTest();
// Aufruf der VBScript-Routine
}
</script>
<SCRIPT LANGUAGE="VBScript">call VBTest()</SCRIPT> <!-- JSTest() wird aktiviert !!! -->
</BODY>
</html>
```

Variablen in JScript und VBScript

Variablen von JScript sind in VBScript verwendbar.

VBScript-Routinen lassen sich in JScript aufrufen.

JScript-Routinen lassen sich in VBScript aufrufen.

Besonderheiten bei der HTML-Codierung

X04='<SCRIPT LANGUAGE="VBScript"> X05=10 end if</SCRIPT>';



liefert Scriptfehler: nicht abgeschlossene Zeichenkonstante: Ursache '</' als Zeichenfolge im String unzulässig

```
// X04='<SCRIPT LANGUAGE="VBScript"> X05=10 end if</SCRIPT>';
```

liefert keinen Scriptfehler, rendert aber Quelltext, obwohl Kommentar kodiert wurde: Ursache '</' als Zeichenfolge im Kommentar unzulässig (dagegen '</' ist zulässig)

Microsoft Encoder für z.B. JScript

Nachfolgend das dynamische Laden einer per MS-Encoder encodierten JS-Datei:

JS-Datei darf nicht die Zeichenkette '<script>' oder '</script>' in jeder Form enthalten,
da sonst als SCRIPT-Tag erkannt
darf sowieso keine Tags enthalten

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
var X00=false;
var X01;
var X02;
var X03='<SCRIPT LANGUAGE="JScript.Encode" SRC="codiert.js"><'+ '/' + 'SCRIPT>';
// < und / immer trennen im String, da sonst Steuerzeichen

// codiert.js aus uncodiert.js entstanden
//
// uncodiert.js hat als Inhalt
//
//      var X00='Encodierte JS-Datei aktiv';
//      function Meldung()
//      {alert(X00);}
//      Meldung();

function init()      // Script-Objekt erzeugen
{X01=document.createElement(X03);X00=(X01!=null);
if(25){X02=document.body.appendChild(X01);X00=(X02!=null);}
if(X00){X00=(X01==X02);} // true so alles korrekt
// wenn true, so wird Meldung() aus codiert.js aktiviert
}
//-->
</SCRIPT>
</HEAD>
<BODY onload='init();'>
</BODY>
</HTML>
```

codiert.js hat folgenden Inhalt: alles genau 1 Zeile

Zeichen ß am Anfang und am Ende sind nur Begrenzer und NICHT Teil des Codes !!
Man beachte die Blanks im und am Ende des Codes !!!

```
ß#@~^WQAAAA==mD~o!Z'BAx1GNb+.Y PBj fIDnk,l3Dr-
Bp@#@&0;x^ObWUPtnV9EULv#@#@&'mVnDDco!T*i)@#@&t+^[E oc#pjRoAAA==^#~@ ß
```

Laden einer JS-Datei per <SCRIPT ... SRC="test.js"></SCRIPT>

wenn das Nachladen per dynamisch erzeugtem Script-Tag erfolgt (dynamisch per createElement() etc.)

und z.B. ein Kommentartext nicht mit // begonnen wurde,

dann kann die readyState-Prüfung des Ladens der test.js unendlich laufen und nie complete erreichen.

Damit gilt: Es wurde trotz Code, der nicht JScript ist, der Fehler nicht erkannt

Besonders fatal ist das für folgende Situation:

Es werden mehrere JS-Dateien nacheinander geladen, wobei diese abhängig sind.

Da die fehlerhafte JS-Datei nicht vollständig geladen wird,
sich aber eine Nachfolge-JS-Datei auf die unvollständig geladene JS beruft,
kann die Nachfolge-JS-Datei ebenfalls kommentarlos endlos laden bzw.
ihrerseits "abstürzen".

Da der IE die JS-Dateien nach dem Laden in ein gemeinsames Dokument als dessen
Inhalt auffasst, also als JS-Dateien-Brei, ist es u. U. schwer, die
Fehlerstelle zu finden, wenn das Laden einer einzelnen JS-Datei mit
Fehler aber ohne Parser-Hinweis erfolgt. (Der IE arbeitet auf Primitiv-Level).

Achtung '</' also Zeichenfolge in 1 gemeinsamen String kann zu Scriptfehler führen !

z.B. var Kette='</SCRIPT>';

jedoch nicht bei var Kette='<'; Kette+='</SCRIPT>';

Attribute



canHaveHTML Sets or retrieves the value indicating whether the object can contain rich HTML markup.

charset Zeichensatz zum Encodieren des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

DEFER defer Sets or retrieves the status of the script.

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

EVENT event Sets or retrieves the event for which the script is written.

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

FOR htmlFor Sets or retrieves the object that is bound to the event script.

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Retrieves a value that indicates the current state of the object.

scopeName Namensbereich des Objektes ermitteln

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen

scrollWidth Scrollbreite vom Objekt ermitteln

sourceIndex Index des Objektes in der Collection document.all ermitteln

SRC src Retrieves the URL to an external file that contains the source code or data.

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

text Retrieves or sets the text of the object as a string.

TYPE type Sets or retrieves the MIME type for the associated scripting engine.

uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

Events

onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)

onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten

removeAttribute Attribute aus einem Objekt entfernen

removeAttributeNode Attribut-Objekt aus einem Objekt entfernen

removeBehavior Behavior aus einem Objekt entfernen

replaceAdjacentText Benachbarten Text ersetzen

setAttribute Wert eines Attributes setzen

setAttributeNode Attribut-Objekt-Knoten einfügen



swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen

layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen

pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen

pixelHeight Höhe des Objektes ermitteln bzw. setzen

pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen

pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen

pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen

pixelWidth Breite des Objektes ermitteln bzw. setzen

posBottom Bottom-Position ermitteln bzw. setzen

posHeight Höhe ermitteln bzw. setzen

posLeft Left ermitteln bzw. setzen

posRight Right ermitteln bzw. setzen

posTop Top ermitteln bzw. setzen

posWidth Breite ermitteln bzw. setzen

text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen

text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen

4.3.2.2.12.9.32. SELECT

Klick auf eine Option schliesst Pull-Down, wenn Cursor dann ausserhalb des Select-Objektes steht, so wird onmouseout ausgelöst

DbClick ist völlig sinnlos: 1. Klick Pulldown öffnen
2. Klick Pulldown schliessen und ev. onmouseout auslösen

wenn DbClick auf Option: 1. Klick Pulldown schliessen und Text der Option im Select anzeigen
2. Klick auf Objekt unter dem Cursor und dann auch onmouseout

Es reicht also onclick aus. Aber: Wenn User blättern will in den Optionen, auch dann wird onclick ausgelöst.

Der onclick-Handler überträgt also sofort den .text der aktuell markieren weil geklickten Option

onclick per fireEvent() auslösen öffnet nicht Selektbereich etc. Es wird nichts getan !!!

Eine Option kann nur per Klick auf diese selektiert werden, wobei damit auch Pull-down

geschlossen wird und, falls der Cursor nicht über Select steht, onmouseout

ausgelöst wird: Letztere Fall dürfte regelmäßig sein, da Pulldown unterhalb

Select steht und mit einklappen wegen klick also der Cursor

auch unterhalb des Selekt steht

Option hat kein onmouseXXX-Event oder onclick-Arten-Event

Damit ist der Klick auf eine Option nicht steuerbar.

Übernahme des Textes einer Option bereits bei onmouseover (User überfährt die Optionen, die auch dabei die Selektionsmarkierung wechseln) ist nicht möglich.

Es ist immer eine Option markiert, sobald Selection geklickt wurde, egal ob User auf eine Option klickt oder nicht.

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen

ALIGN align Sets or retrieves how the object is aligned with adjacent text.

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist

DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

DISABLED disabled Sets or retrieves a value that indicates whether the user can interact with the object.

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

form Retrieves a reference to the form that the object is embedded in.

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

length Sets or retrieves the number of objects in a collection.

MULTIPLE multiple Sets or retrieves the Boolean value indicating whether multiple items can be selected from a list.



NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Retrieves the document object associated with the node.

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Status des Objektes ermitteln

recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

scrollHeight Scrollhöhe vom Objekt ermitteln

scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen

scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen

scrollWidth Scrollbreite vom Objekt ermitteln

selectedIndex Sets or retrieves the index of the selected option in a select object.

SIZE size Sets or retrieves the number of rows in the list box.

sourceIndex Index des Objektes in der Collection document.all ermitteln

STYLE style inline-Style des Objektes setzen

TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

type Retrieves the type of select control based on the value of the MULTIPLE attribute.

uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen

VALUE value Sets or retrieves the value which is returned to the server when the form control is submitted.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird

onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten

onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird

onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird

onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält

onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird

onbeforeupdate Event erzeugt direkt vor dem Update der Daten

onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)

onchange Fires when the contents of the object or selection have changed.

onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschneitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird

onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde

onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren

onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird

onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird

onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird

onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert

onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt

onmouseout Event erzeugt, sobald die Maus das Objekt verlässt

onmouseover Event erzeugt, sobald Maus das Objekt betritt



onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

Methoden

add Adds an element to the areas, controlRange, or options collection.
 addBehavior Dem Objekt ein Behavior hinzufügen
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 remove Removes an element from the collection.
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen
 urns Retrieves a collection of all objects to which a specified behavior is attached.

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 ACCELERATOR accelerator Tastaturkürzel setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen



font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
fontWeight Sets or retrieves the numeric weight of the font of the object.
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for an element.
max-width maxWidth Sets or retrieves the maximum width for an element.
min-height minHeight Sets or retrieves the minimum height for an element.
min-width minWidth Sets or retrieves the minimum width for an element.
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus
text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objekumgebung wird z.T. automatisch angepasst)
width width Breite eines Objektes ermitteln bzw. setzen
word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.33. SPAN

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
CLASS className Klassenname des Objektes ermitteln bzw. setzen
clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
DATAFORMATAS dataFormatAs Art der Renders von Daten im Objekt ermitteln bzw. setzen
DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
hasMedia Status (booealan) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist



HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
 systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mousevoer) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermittelt (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Selektierbarkeit des Objektes ausschalten

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist



ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
onkeydown Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
onkeydown Event erzeugt, sobald eine gedrückte Taste losgelassen wird
onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
onmouseover Event erzeugt, sobald Maus das Objekt betritt
onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
onmousewheel Event erzeugt, wenn Mausrad gedreht wird
onmove Event erzeugt, wenn Objekt bewegt wird
onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
contains Ermitteln ob Objekt andere Objekte enthält
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
doScroll Erzeugt Klick auf die Scrollbar des Objektes
dragDrop Drag-Event für das Objekt erzeugen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
getAdjacentText Benachbarten Textstring liefern
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
getExpression Ausdruck für eine Eigenschaft liefern
hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
insertAdjacentElement Nachbar-Objekt einfügen
insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
insertAdjacentText Text als Nachbar-Text einfügen
insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
releaseCapture Maus-Event-Überwachung zum Objekt entfernen
removeAttribute Attribute aus einem Objekt entfernen
removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
removeBehavior Behavior aus einem Objekt entfernen
removeChild Kindknoten aus einem Objekt entfernen
removeExpression Ausdruck aus einer Eigenschaft entfernen
removeNode Objektknoten aus Dokument-Hierarchie entfernen
replaceAdjacentText Benachbarten Text ersetzen
replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
replaceNode Knoten durch anderen Knoten ersetzen
scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)



setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
 overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
 overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen



pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 right right Right ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, an der Fluss erfolgt
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 text-decorationBlink Textblinker ein aus
 text-decorationLineThrough Text durchstreichen ein aus
 text-decorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 text-decorationOverline Linie über Text zeichnen ein aus
 text-decorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können
 width width Breite eines Objektes ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.34. STYLE (STYLE-Attribut oder .style, nicht styleSheet)

Styles gibt es in diversen Formen, z.B. als

CSS-Datei (siehe styleSheet)

STYLE im HEAD

STYLE-Attribut bzw. Eigenschaft .style eines Objektes (inline Style)

Inline-Styles (STYLE-Attribut bzw. .style) sind objektspezifisch implementiert.

Die Styleerzeugung per HTML hängt von der HTML-Implementation ab, die sich von der per .style unterscheiden kann.

Die Kodierung von Style-Folgen ist per HTML möglich.

Die Styleerzeugung per Script sollte nur per .style erfolgen.

Die Kodierung von Style-Folgen per .style ist nicht möglich.

Die Style-Kodierung unterscheiden sich

Bspl:	background-color	in der CSS-Regel
		im STYLE-Attribut
	backgroundColor	in .style

Styles können mehrere Styles umfassen: Listenform des Style-Wertes

Bspl.:

background im CSS, STYLE-Attribut und im .style

```

tagname { background : sBackground }
STYLE="background:sBackground"
style.background [ = sBackground ]
  
```

sBackground Liste aus Stylewerten, die mit Leerzeichen getrennt sind

color Farbe des Hintergrundes (wie bei .style.backgroundColor)

image Hintergrundbild (wie bei .style.backgroundImage)

repeat Wiederholung des Hintergrundbildes (wie bei .style.backgroundRepeat property)

attachment Attachmentwert (wie bei .style.backgroundAttachment)

position Positionswert (wie bei .style.backgroundPosition)

Standardwert ist 'transparent none repeat scroll 0% 0%'

Beispiel:

<STYLE>



```
.style1 { background:beige url(sphere.jpg) no-repeat top center }
.style2 { background:ivory url(sphere.jpeg) no-repeat bottom right }
</STYLE>
</HEAD>
<BODY>
<SPAN onmouseover="this.className='style1'" onmouseout="this.className='style2'">. . . </SPAN>

<SPAN onclick="this.style.background='beige url(sphere.jpeg) no-repeat top center'">. . . </SPAN>
```

Es gibt Styles, die haben ein zusätzliches eigenes Attribut im HTML-Code (also nicht nur die Kodierung im STYLE-Attribut)

Bsp:

```
BODY { background [ = sURL ] }
<BODY BACKGROUND = sURL... >
<BODY STYLE='background-image:SURL">
document.body.style.backgroundImage=sURL;
```

Es ist möglichst per .style zu kodieren.

DHTML-Behavior als Styles:

Behavior können nur per CSS oder style kodiert werden, sind aber Objekte.

```
ELEMENT { behavior : sBehavior }
<ELEMENT STYLE="behavior:sBehavior" >
object.style.behavior [ = sBehavior ]

sBehavior String als Liste aus Leerzeichengetrennten Strings der Form 'url(sLocation)'
sLocation absolute oder relative Url des DHTML-Behavior in einer htc-Datei
HTML (DHTML) behavior, where sLocation is an absolute or relative URL.
z.B. url(a1.htc) url(a2.htc) ...

url(#objID) ID eines OBJECT-Tags
Objekt enthält Binärcodes des Behavior
url(#default#behaviorName)
Standard Behaviors des IE
behaviorName steht für den Bezeichner des Behaviors

lesen und schreiben
nur lesen bei currentStyle
```

Beispiel 1

```
<UL>
<LI STYLE="behavior:url(ul.htc) url(hilite.htc)">HTML</LI>
<UL>
<LI><tla rid="tla_ie4"/> authoring tips</LI>
:
</UL>
</UL>
```

Beispiel 2

```
<style>
.CollapsingAndHiliting { behavior:url(ul.htc) url(hilite.htc) }
</style>

<UL>
<LI CLASS="CollapsingAndHiliting">HTML</LI>
<UL>
<LI><tla rid="tla_ie4"/> authoring tips</LI>
:
</UL>
</UL>
```

Beispiel 3

```
<SCRIPT>
function window.onload()
{
idTopic1.style.behavior = "url(ul.htc) url(hilite.htc)";
}
</SCRIPT>

:

<UL>
<LI ID=idTopic1>HTML Authoring</LI>
```



```

<UL>
  <LI><tla rid="tla_ie4"/> authoring tips</LI>
:
</UL>

```

Folgende Objekte haben .style.behavior:

A, ABBR, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, currentStyle, CUSTOM, DD, defaults, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit,

INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID,

NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, runtimeStyle, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, style, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, WBR, XML, XMP

Folgende Standard-Behavior-Objekte des IE gibt es:

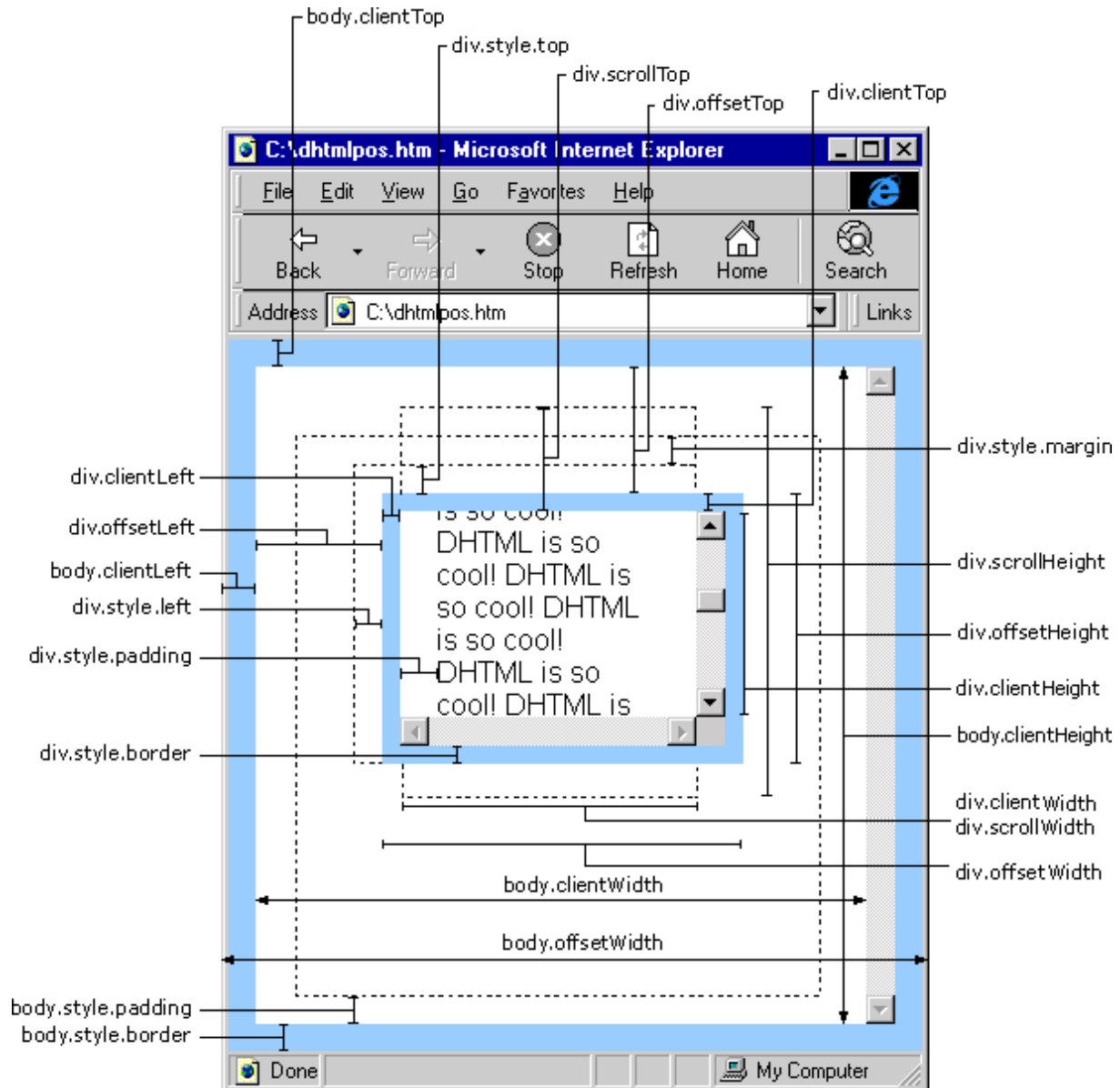
anchorClick	Dateiordner-Ansicht im Dokument
anim	Instanz des Microsoft DirectAnimation-Viewer zum Rendern von DirectAnimation-Objektes und DirectAnimation-Sounds
clientCaps	Informationen zu Features des IE
download	Download-Objekt mit Funktionsaufruf bei Downloadende
homePage	Informationen zur Webseite

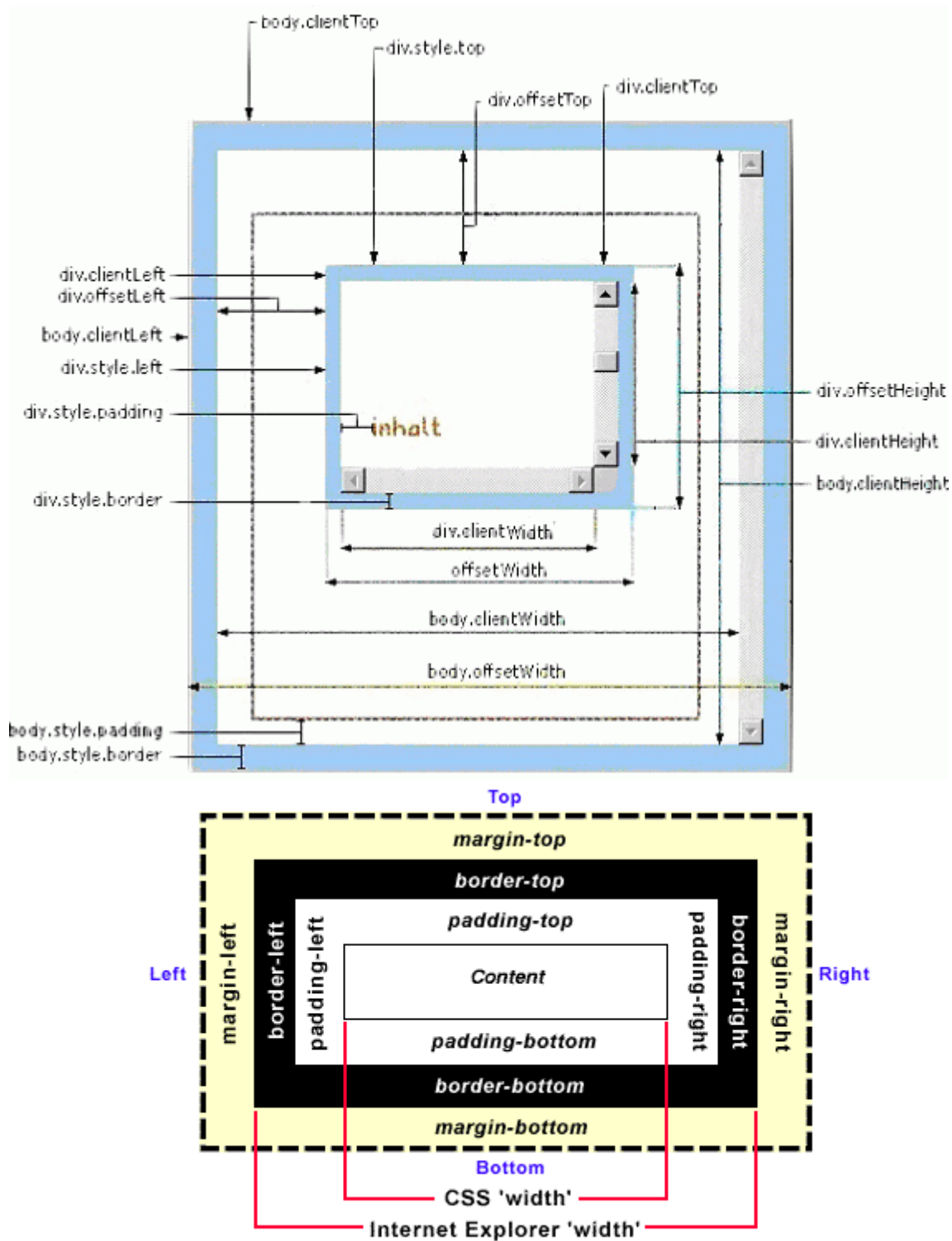
Beispiel: `<a href="#"`
`onClick="this.style.behavior='url(#default#homepage)';`
`this.setHomePage('http://www.test.de');"`
`>`
 Machen Sie diese Seite zu Ihrer Startseite
``

httpFolder	Scripting-Features für Dateiordner-Ansicht im Dokument
mediaBar	Basic-User Interface für Mediawiedergabe im Browser
saveFavorite	Favorit zum Dokument erzeugen
saveHistory	History sichern
saveSnapshot	Webseite speichern
userData	Userdaten speichern im userData-Bereich des Browsers

Begriffe aus Style:







Style und CSS-Beeinflussung durch !DOCTYPE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!-- IE-spezifischer DOCTYPE, der nicht zu ändern ist -->
<!-- niemals ... DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd" -->
<!-- da sonst IE nur noch CSS-Standard laut Url nutzt und keinen IE-eigenen CSS mehr nutzt -->
<!-- niemals ... DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Strict//EN" .... mit oder ohne "http ...." -->
<!-- da HTML-Attribute verwendet sein können, die veraltet sind, aber für die es keinen Style-Ersatz gibt -->
```

Objekte mit STYLE-Attribut (bzw. .style) sind

A ACRONYM ADDRESS APPLET AREA B BIG BLOCKQUOTE BODY BR BUTTON CAPTION
CENTER CITE CODE COL COLGROUP CUSTOM DD DEL DFN DIR DIV DL DT EM EMBED



FIELDSET FONT FORM hn HR I IFRAME IMG INPUT type=button INPUT type=checkbox
 INPUT type=file INPUT type=hidden INPUT type=image INPUT type=password INPUT type=radio
 INPUT type=reset INPUT type=submit INPUT type=text INS ISINDEX KBD LABEL LEGEND LI
 LISTING MAP MARQUEE MENU OBJECT OL P PLAINTEXT PRE Q RT RUBY S SAMP SELECT
 SMALL SPAN STRIKE STRONG SUB SUP TABLE TBODY TD TEXTAREA TFOOT TH THEAD TR
 TT U UL VAR XMP

Objekte mit Style-eigenschaft .visibility sind

A ADDRESS APPLET B BIG BLOCKQUOTE BODY BUTTON CAPTION CENTER CITE CODE COL
 COLGROUP currentStyle CUSTOM DD defaults DFN DIR DIV DL DT EM EMBED FIELDSET FORM hn
 HR HTML I IFRAME IMG INPUT type=button INPUT type=checkbox INPUT type=file INPUT type=image
 INPUT type=password INPUT type=radio INPUT type=reset INPUT type=submit INPUT type=text
 ISINDEX KBD LABEL LEGEND LI LISTING MARQUEE MENU NOBR OBJECT OL P PRE S
 SAMP SELECT SMALL SPAN STRIKE STRONG style SUB SUP TABLE TBODY TD TEXTAREA TFOOT
 TH THEAD TR TT U UL VAR XMP

style.visibility hat NICHT den Standardwert
 'inherit', denn den gibt es nicht
 " egal ob Elternobjekt sichtbar oder nicht

eine Style-Eigenschaft ist nicht auslesbar, wenn diverse Wertarten in die Eigenschaft hinterlegbar sind z.B. String oder numerisch
 Bsp: if(zeiger.style.top>=0) ... liefert immer false, da top divers belegbar ist

Zur Positionierung eines Objektes kann benutzt werden (falls implementiert)

zeiger_auf_objekt.style.position='absolute'; bzw. zeiger_auf_objekt.style.position='relative';
 zeiger_auf_objekt.style.left=wert
 zeiger_auf_objekt.style.top=wert

zeiger_auf_objekt entstammt z.B.
 aus createElement()
 oder aus dem ID-Attribut
 oder aus einer der getElementByXXXX()-Funktionen
 .left und .top sind lesbar und schreibbar mit diversen Wertarten z.B. 8px oder 8 etc.

Folgende Objekte haben .style.position

A, ADDRESS, APPLET, B, BDO, BIG, BLOCKQUOTE, BUTTON, CENTER, CITE, CODE, currentStyle,
 CUSTOM,
 DD, defaults, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT
 type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT
 type=radio,
 INPUT type=reset, INPUT type=submit, INPUT type=text, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING,
 MARQUEE, MENU, OBJECT, OL, P, PRE, RUBY, runtimeStyle, S, SAMP, SELECT, SMALL, SPAN, STRIKE,
 STRONG, style, SUB, SUP, TABLE, TD, TEXTAREA, TH, TR, TT, U, UL, VAR, XMP

Folgende Objekte haben .style.left

A, ADDRESS, APPLET, B, BIG, BLOCKQUOTE, BUTTON, CENTER, CITE, CODE, currentStyle, CUSTOM, DD,
 defaults, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT
 type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT
 type=radio,
 INPUT type=reset, INPUT type=submit, INPUT type=text, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING,
 MARQUEE, MENU, OBJECT, OL, P, PRE, runtimeStyle, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG,
 style, SUB, SUP, TABLE, TEXTAREA, TT, U, UL, VAR, XMP

Folgende Objekte haben .style.top

A, ADDRESS, APPLET, B, BIG, BLOCKQUOTE, BUTTON, CENTER, CITE, CODE, currentStyle, CUSTOM, DD,
 defaults, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT
 type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT
 type=radio,
 INPUT type=reset, INPUT type=submit, INPUT type=text, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING,
 MARQUEE, MENU, OBJECT, OL, P, PRE, runtimeStyle, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG,
 style, SUB, SUP, TABLE, TEXTAREA, TT, U, UL, VAR, XMP

Für Positionsangaben innerhalb des Fensters des Dokumentes, also innerhalb des Client, kann lesend benutzt werden

document.body.clientHeight als Höhe
 document.body.clientWidth als Breite

wobei in den Werten nicht beachtet werden:
 margin, border, scroll bar, caption bar, menu bar, toolbars, status bars
 wobei die Maximalwerte von der Browserversion und deren Symbolleisten etc. abhängen:



Hintergrundbild per .backgroundImage
für document

```
document.body.style.backgroundImage='url('+ 'test.gif' + ')';
// Belegung geht NUR per url()
// Achtung: url() existiert nicht als Zeiger
// var Kette=url() geht also nicht
// ist interne Funktion

für Popup (createPopup)
popupzeiger.document.body.style.backgroundImage='url('+ 'test.gif' + ')';
mit identischer Wirkung ist
popupzeiger.document.body.innerHTML='<IMG SRC="test.gif">';
Achtung: transparente GIF-Dateien werden immer nicht-transparent gerendert !
animierte GIF-Dateien werden nicht immer animiert gerendert !
```

Zeigerverwendung bei .style

```
var X_Zeiger=object.style; // Zeiger auf Style
var X_Zeiger_innerHTML=object.style.innerHTML;

X_Zeiger_innerHTML='<DIV></DIV>'; // wird NICHT gerendert
X_Zeiger.innerHTML='<DIV></DIV>'; // wird gerendert !
```

Die Eigenschaft von .style kann nicht per Zeiger adressiert werden

Bsp.: .style.visibility ist der Zeiger auf den Wert von visibility
und nicht Zeiger auf die Style-Eigenschaft visibility

Das Kopieren von Style per Zeiger ist nicht möglich:

```
alert(objekt1.style); zeigt an "[object]"
also ist .style ein Zeiger
```

Bsp.: objekt1.style=objekt2.style;

Es erfolgt Fehlermeldung: Member nicht gefunden.
Würde kopiert werden, so würde DOM durcheinander gebracht, weil
Objektorientierung dann nicht mehr gilt:
Daten per Zeiger in ein fremdes Objekt einfügen --> Objekt wäre nicht mehr konsistent

Als Lösung geht nur:

```
Styles als Strings verwalten
z.B. var test='cursor="hand";color="white";'
// test kann kopiert werden
// nach kopieren den Style wie folgt zuweisen
var Feld=test.split(';');
var Kette='objekt1.style.';
Kette+=Feld[0];
Kette+='.';
eval(Kette); // weist cursor zu
// für Feld[1] analog bzw. alles als Schleife
```

Attribute für STYLE im HEAD

disabled Sets or retrieves whether a style sheet is applied to the object.
ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
innerHTML Retrieves the HTML between the start and end tags of the object.

Events für STYLE im HEAD

onerror Fires when an error occurs during object loading.
onreadystatechange Fires when the state of the object has changed.

Methoden für STYLE im HEAD

addBehavior Dem Objekt ein Behavior hinzufügen
dragDrop Drag-Event für das Objekt erzeugen
removeBehavior Behavior aus einem Objekt entfernen
removeNode Removes the object from the document hierarchy.

Styles für STYLE im HEAD

MEDIA media Sets or retrieves the media type.
TYPE type Retrieves the Cascading Style Sheets (CSS) language in which the style sheet is written

Attribute für STYLE eines Objektes (STYLE-Attribut oder .style)

onOffBehavior Retrieves an object indicating whether the specified Microsoft® DirectAnimation® behavior is running.

Events für STYLE eines Objektes (STYLE-Attribut oder .style)

keine



Methoden für STYLE eines Objektes (STYLE-Attribut oder .style)

getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribut-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 removeAttribute Attribut aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setExpression Ausdruck setzen

Styles für STYLE eines Objektes (STYLE-Attribut oder .style)

ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-collapse borderCollapse Sets or retrieves a value that indicates whether the row and cell borders of a table are joined in a single border or detached as in standard HTML.
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cssText Sets or retrieves the persisted representation of the style rule.
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 height height Objekthöhe ermitteln bzw. setzen
 ime-mode imeMode Sets or retrieves the state of an Input Method Editor (IME).
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
 left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 list-style listStyle Sets or retrieves up to three separate listStyle properties of the object.
 list-style-image listStyleImage Sets or retrieves a value that indicates which image to use as a list-item marker for the object.
 list-style-position listStylePosition Sets or retrieves a variable that indicates how the list-item marker is drawn relative to the content of the object.



list-style-type listStyleType Sets or retrieves the predefined type of the line-item marker for the object.

margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen

margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen

margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen

margin-right marginRight Dicke von right-margin ermitteln bzw. setzen

margin-top marginTop Höhe von top-margin ermitteln bzw. setzen

min-height minHeight Sets or retrieves the minimum height for an element.

overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf

overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf

overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf

padding padding Padding ermitteln bzw. setzen

padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen

padding-left paddingLeft Padding-left ermitteln bzw. setzen

padding-right paddingRight Padding-right ermitteln bzw. setzen

padding-top paddingTop Padding-Top ermitteln bzw. ersetzen

page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf

page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf

pixelBottom Bottom-Position der unteren Kante des Objektes ermitteln bzw. setzen

pixelHeight Höhe des Objektes ermitteln bzw. setzen

pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen

pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen

pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen

pixelWidth Breite des Objektes ermitteln bzw. setzen

posBottom Bottom-Position ermitteln bzw. setzen

posHeight Höhe ermitteln bzw. setzen

position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen

posLeft Left ermitteln bzw. setzen

posRight Right ermitteln bzw. setzen

posTop Top ermitteln bzw. setzen

posWidth Breite ermitteln bzw. setzen

right right Right ermitteln bzw. setzen

ruby-align rubyAlign Sets or retrieves the position of the ruby text specified by the rt object.

ruby-overhang rubyOverhang Sets or retrieves the position of the ruby text specified by the rt object.

ruby-position rubyPosition Sets or retrieves the position of the ruby text specified by the rt object.

scrollbar-3dlight-color scrollbar3dLightColor Farbe der oberen und linken Kante von Scroobar und Scroll-Pfeile ermitteln bzw. setzen

scrollbar-arrow-color scrollbarArrowColor Farbe der Scrol-Pfeile ermitteln bzw. setzen

scrollbar-base-color scrollbarBaseColor Farbe der Hauptelement der Scrollbar, Scrollbox und Scrollpfeile ermitteln bzw. setzen

scrollbar-darkshadow-color scrollbarDarkShadowColor Schattenfarbe der Scrollbar ermitteln bzw. setzen

scrollbar-face-color scrollbarFaceColor Farbe der Scrollbox und Scrollpfeile ermitteln bzw. setzen

scrollbar-highlight-color scrollbarHighlightColor Farbe der oberen und linken Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen

scrollbar-shadow-color scrollbarShadowColor Farbe der unteren und der rechten Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen

scrollbar-track-color scrollbarTrackColor Farbe der Trackelemente der Scrollbar ermitteln bzw. setzen

float styleFloat Seite des Objektes ermitteln bzw. setzen, and der Fluss erfolgt

table-layout tableLayout Sets or retrieves a string that indicates whether the table layout is fixed.

text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)

text-align-last textAlignLast Ausrichtung der letzten Text-Linie im Objekt ermitteln bzw. setzen

text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen

text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen

textDecorationBlink Textblinken ein aus

textDecorationLineThrough Text durchstreichen ein aus

textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus

textDecorationOverline Linie über Text zeichnen ein aus

textDecorationUnderline Linie unter Text zeichnen ein aus

text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen

text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen

text-kashida-space textKashidaSpace Layout kashida ermitteln bzw. setzen

text-overflow textOverflow render ellipses(...) für Text-Overflow ein aus

text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen

text-underline-position textUnderlinePosition Position der Untertrichlinie ermitteln bzw. setzen

top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)

unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen

vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.

visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)

white-space whiteSpace Status ermitteln bzw. setzen, ob Textlinien per white-space umgebrochen werden können

width width Breite eines Objektes ermitteln bzw. setzen

word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen

word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen

word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus

writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen

z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)

zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.35. styleSheet (nicht STYLE-Attribut oder .style)

Begriffe für Style siehe Objekt style



styleSheet ist eine CSS-Regel (outline-Style)

Beispiel

```
<STYLE>
BODY {background-color: #CFCFCF;}
@import url("otherStyleSheet.css");
</STYLE>
```

Durch JScript kann eine CSS-Regel eventuell per createElement() erzeugt werden.

Anstelle CDD-Regeln kann auch inline-Style per .style verwendet werden.

Die Style-Kodierung unterscheiden sich

Bspl:	background-color	in der CSS-Regel im STYLE-Attribut
	backgroundColor	in .style

Attribute

canHaveHTML Sets or retrieves the value indicating whether the object can contain rich HTML markup.

disabled Sets or retrieves whether a style sheet is applied to the object.

href Sets or retrieves the URL of the linked style sheet.

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

owningElement Retrieves the next object in the HTML hierarchy.

parentStyleSheet Retrieves the style sheet that imported the current style sheets.

readOnly Retrieves whether the rule or style sheet is defined on the page or is imported.

TITLE title Sets or retrieves the title of the style sheet.

type Retrieves the Cascading Style Sheets (CSS) language in which the style sheet is written.

Events

keine

Methoden

addImport Adds a style sheet to the imports collection for the specified style sheet.

addPageRule Creates a new page object for a style sheet.

addRule Creates a new rule for a style sheet.

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

removeRule Deletes an existing style rule for the styleSheet object, and adjusts the index of the rules collection accordingly.

Styles

cssText Sets or retrieves the persisted representation of the style rule.

MEDIA media Sets or retrieves the media type.

text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen

4.3.2.2.12.9.36. TABLE

Tabelle erzeugen per createElement()

Elemente der Tabelle können sein (Tagnamen)

TR	Zeile bestehend aus optionalen Spalten (TD)
TD	Spalte in einer Zeile (TR)
COLGROUP	Gruppierte Spalten (TD)

kann keine TD enthalten, da sonst unwirksam

Besonderheit beim codieren in HTML Beispiel:

```
<TABLE BORDER="2" RULES="groups">
  <COLGROUP SPAN="2" STYLE="color:red"></COLGROUP>
  <COLGROUP STYLE="color:blue"></COLGROUP>
  <TR>
    <TD>Zeile 1 This column is in the first group.</TD>
    <TD>Zeile 1 This column is in the first group.</TD>
    <TD>Zeile 1 This column is in the second group.</TD>
  </TR>
  <TR>
    <TD>Zeile 2 This column is in the first group.</TD>
    <TD>Zeile 2 This column is in the first group.</TD>
    <TD>Zeile 2 This column is in the second group.</TD>
  </TR>
</TABLE>
```

erste COLGROUP hat 2 Spalten (TD)

zweite COLGROUP hat unbekannte Zahl von Spalten (TD)

Der Browser ordnet die erste 2 TD der jeweiligen TR der ersten COLGROUP zu
alle anderen TD der jeweiligen TR der zweiten COLGROUP zu

Tabelle erzeugen per createElement() Schritt 1

```
var X00=document.createElement("TABLE");
```



```
var TabellenZeiger=document.body.appendChild(X00);
```

Tabelle erzeugen per createElement() Schritt 2 - Variante 1 (Methoden von TABLE)

```
Tabellenelement erzeugen am Beispiel TR
oTR = TabellenZeiger.insertRow(); // automatisch append
```

Folgend Funktionen gibt es:

Es gibt keine Funktion für COLGROUP

id_head=id_table.createTHead();	liefert Zeiger auf TH oder null mehrere TH erzeugbar
id_table.deleteTHead();	zuletzt erzeugten TH löschen, liefert nichts es gibt keine Index da keine Collection verfügbar (ausser document.all)
id_TR=id_table.insertRow([iRowIndex]);	liefert Zeiger auf TR oder null iIndex -1 Standard für append an rows-Collection zur Tabelle wenn >=0 so Element an aktueller Indexpos in Collection rows eingebaut Achtung: Index sollte lückenlos sein wegen Konsistenz des Objektes
id_table.deleteRow([iRowIndex]);	löscht TR iRowIndex Index ab 0 des TR in rows-Collection zu Tabelle Standard letzte TR in rows-Collection
id_TD=id_TR.insertCell([iCellsIndex]);	liefert Zeiger auf TD oder null iIndex -1 Standard für append an cells-Collection zum TR wenn >=0 so Element an aktueller Indexpos in Collection cells eingebaut Achtung: Index sollte lückenlos sein wegen Konsistenz des Objektes
id_TR.deleteCell([iCellsIndex]);	löscht TD iCellsIndex Index ab 0 des TD in cells-Collection Standard letzte TD in cells-Collection zu TR
id_TFoot=id_table.createTFoot();	liefert Zeiger auf Foot oder null mehrere Foot erzeugbar
id_tabelle.deleteTFoot();	zuletzt erzeugten Foot löschen, liefert nichts es gibt keine Index da keine Collection verfügbar (ausser document.all)
id_Caption=id_table.createCaption();	liefert Zeiger auf Caption oder null mehrere erzeugbar aber nur die ERSTE Caption löschar !
id_table.deleteCaption();	löscht NUR ZUERST erzeugtes Caption es gibt keine Index da keine Collection verfügbar (ausser document.all)

Beispiel für Erzeugung einer Tabelle

```
// +++++ Datenfelder indexsynchron und identische Länge
var X00=new Array
(true, // 1. TR, 1. Element MUSS TR sein
 false,false,false, // alle TD der 1. TR
 true, // 2. TR
 true, // 3. TR
 false // alle TD der 3. TR
);

var X00a=new Array // .innerHTML
(", // 1. TR kein Inhalt da .innerText und .innerHTLM nicht belegbar
 'a','b','c', // alle TD der 1. TR
 ",
 ",
 '<B>d</B>' // alle TD der 3. TR
);

// +++++ Zeigerfeld der Tabellenelemente erzeugen
var X01=new Array();
// +++++ Tabelle erzeugen per DOM
var X02=document.createElement('TABLE');
var X03=document.body.appendChild(X02);
// +++++ Caption erzeugen
var X99=X03.createCaption();
X99.innerHTML='<B>Tabellenkopf</B>';
// X00.valign='top' oder 'bottom' funktioniert nicht, immer 'top' automatisch also über Tabelle
```



```
//          innerHTML-Breite per Schrift beeinflusst Tabellenbreite beim rendern !
// ++++++ Tabellenelemente erzeugen
var X04=0;
var X05=X00.length;           // Länge von X00 und X00a
var X06=false;                 // Feldelement aus X00
var X07=-1;                    // Index TR, muss -1 sein
var X08=-1;                    // Index TD in TR, muss -1 sein
for(X04=0;X04<X05;X04++)      // Datenfelder abklappern
{ X06=X00[X04];                // Flag auf TD bzw. TR
  if(X06)                       // TR
  { X07++;                      // nächstes TR einstellen
    X01[X07]=X03.insertRow();   // TR in Tabelle einfügen
    X08=-1;                     // init Index TD
  }
  else                           // TD zum aktuellen TR
  { X08++;                      // nächstes TD einstellen
    X01[X07][X08]=X01[X07].insertCell();
    X01[X07][X08].innerHTML=X00a[X04];
  }
}
```

Tabelle erzeugen per createElement() Schritt 2 - Variante21 (Methoden des DOM)

Erzeugung per DOM und Objekte THEAD, TBODY, TFOOT, CAPTION

Beispiel für TBODY

```
var X00=document.createElement('TBODY');
var TBODYObjekt=TabellenZeiger.appendChild(X00);
```

Tabelle darf mehrere TBODY haben

Beispiel für TR im TBODY (TD analog)

```
var X00=document.createElement('TR');
var TRObjekt=TBODYObjekt.appendChild(X00);
```

THEAD darf enthalten	TR (auch mehrere) und damit auch TD
TBODY darf enthalten	TR (auch mehrere) und damit auch TD
	mehrere TBODY pro Tabelle zulässig
TFOOT darf enthalten	TR (auch mehrere) und damit auch TD
CAPTION	Titel der Tabelle standardgemäß über dem Tabellenbeginn
	Positionierung per .valign='top'; // Standard
	.valign='bottom';
	mehr gibts nicht

.innerText und .innerHTML

für TR-Objekt	nur lesbar
für TD-Objekt	lesen und schreiben

Collectionen zu Tabellenelementen TR und TD

Tabelle hat Collection	
row aller TR der Tabelle	id_tabelle.rows
tBodies aller TBODY der Tabelle	id_tabelle.tBodies

jede TR hat eigene Collection cells aller TD zum TR	id_TR.cells
---	-------------

Collectionen

- haben Eigenschaft .length
- werden gefüllt in Reihenfolge der Erzeugung der Tabellenelemente
- Index des Objektes in der Collection ist ab 0

id_objekt.sourceIndex	liefert Index ab 0 aus Collection document.all (nur lesen)
id-Objekt	TD, TR, TH, Tbody, TFoot, Caption

TR und TD haben folgende Eigenschaften bezüglich zugehöriger Collection:

id_TR.rowIndex	liefert Index ab 0 aus row-Collection der Tabelle
id_TD.cellIndex	liefert Index ab 0 aus cells-Collection des zugehörigen TR-Objektes

Tabelle mit fixen Dimensionen
Beispiel




```
TABLE STYLE="table-layout:fixed" WIDTH=600>
<COL WIDTH=100><COL WIDTH=300><COL WIDTH=200>
<TR HEIGHT=20>
<TD>...</TD><TD>...</TD><TD>...</TD>
</TR>
```

```
zeiger_auf_tabelle.style.tableLayout [ = sLayout ]
```

sLayout	'auto'	Standard
		Spaltenbreite automatisch angepasst
	'fixed'	100%
		oder Spaltenbreite ermitteln wenn Breitenangaben kodiert
		Summe aller Spaltenbreiten
		oder Breite der 1. Zeile

Attribute

ACCESSKEY Kurztastenkombination ermitteln bzw. setzen

ALIGN align Sets or retrieves a value that indicates the table alignment.

ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen

BACKGROUND background Sets or retrieves the background picture tiled behind the text and graphics in the object.

BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen

BGColor bgColor Diese Eigenschaft ist nicht mehr verwendbar !

BORDER border Sets or retrieves the width of the border to draw around the object.

BORDERCOLOR borderColor Sets or retrieves the border color of the object.

borderColorDark Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.

borderColorLight Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.

canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf

canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf

caption Retrieves the caption object of the table.

CELLPADDING cellPadding Sets or retrieves the amount of space between the border of the cell and the content of the cell.

CELLSPACING cellSpacing Sets or retrieves the amount of space between cells in a table.

CLASS className Klassenname des Objektes ermitteln bzw. setzen

clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)

clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters

clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters

clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)

COLS cols Sets or retrieves the number of columns in the table.

DATAPAGESIZE dataPageSize Sets or retrieves the number of records displayed in a table bound to a data source.

DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen

DIR dir Lesereichtung des Objektes ermitteln bzw. setzen

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

FRAME frame Sets or retrieves the way the border frame around the table is displayed.

hasMedia Status (booealan) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist

Height height Objekthöhe ermitteln bzw. setzen

HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML Retrieves the HTML between the start and end tags of the object.

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

LANGUAGE language Scriptsprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft

offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln

offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln

onOffBehavior Referenz auf Objekt ermitteln, das anzeigt, ob Microsoft DirectAnimation-Behavior aktiv ist

outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen

outerText Text-Kontent um das Objekt ermitteln bzw. setzen

ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln



parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
readyState Retrieves a value that indicates the current state of the object.
RULES rules Sets or retrieves which dividing lines (inner borders) are displayed.
scopeName Namensbereich des Objektes ermitteln
scrollHeight Scrollhöhe vom Objekt ermitteln
scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
scrollWidth Scrollbreite vom Objekt ermitteln
sourceIndex Index des Objektes in der Collection document.all ermitteln
STYLE style inline-Style des Objektes setzen
SUMMARY summary Sets or retrieves a description and/or structure of the object.
SYNCMaster syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung
systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
tfoot Retrieves the tFoot object of the table.
thead Retrieves the tHead object of the table.
TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
onbeforeactivate Event erzeugt direkt bevor Objekt aktives Objekt wird
onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschneitten und in die Zwischenablage von Windows kopiert wird
ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
onfocusin Event erzeugt direkt bevor Objekt den Fokus erhält
onfocusout Event erzeugt direkt nachdem Objekt den Fokus erhält
onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
onmouseover Event erzeugt, sobald Maus das Objekt betritt
onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
onmousewheel Event erzeugt, wenn Mausrad gedreht wird
onmove Event erzeugt, wenn Objekt bewegt wird
onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
onresize Event erzeugt, wenn sich die Größe (Dimension) des Objektes ändert



onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 createCaption Creates an empty caption element in the table.
 createTFoot Creates an empty tFoot element in the table.
 createTHead Creates an empty tHead element in the table.
 deleteCaption Deletes the caption element and its contents from the table.
 deleteRow Removes the specified row (tr) from the element and from the rows collection.
 deleteTFoot Deletes the tFoot element and its contents from the table.
 deleteTHead Deletes the tHead element and its contents from the table.
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 firstPage Displays the first page of records in the data set to which the table is bound.
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 insertRow Creates a new row (tr) in the table, and adds the row to the rows collection.
 lastPage Displays the last page of records in the data set to which the table is bound.
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 moveRow Moves a table row to a new position.
 nextPage Displays the next page of records in the data set to which the table is bound.
 normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten
 previousPage Displays the previous page of records in the data set to which the table is bound.
 refresh Refreshes the content of the table. This might be necessary after a call to a method such as removeRule, when the page does not automatically reflow.
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut-Objekt aus einem Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Objekt-Knoten einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen



background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
border-collapse borderCollapse Sets or retrieves a value that indicates whether the row and cell borders of a table are joined in a single border or detached as in standard HTML.
border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
border-left borderLeft Linker Rahmen
border-left-color borderLeftColor Linker Rahmen
border-left-style borderLeftStyle Linker Rahmen
border-left-width borderLeftWidth Linker Rahmen
border-right borderRight Rechter Rahmen
border-right-color borderRightColor Rechter Rahmen
border-right-style borderRightStyle Rechter Rahmen
border-right-width borderRightWidth Rechter Rahmen
border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
direction direction Leserichtung ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
padding padding Padding ermitteln bzw. setzen
page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen
float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
table-layout tableLayout Sets or retrieves a string that indicates whether the table layout is fixed.
text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)



text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
 text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-break wordBreak Zeilenumbruch in wörtern bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.37. CAPTION

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ALIGN align Sets or retrieves the alignment of the caption or legend.
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 ownerHTML HTML-Kontext um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Status des Objektes ermitteln
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)



UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALIGN vAlign Sets or retrieves whether the caption appears at the top or bottom of the table.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln



getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen
 border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
 border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
 border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
 border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
 border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
 border-left borderLeft Linker Rahmen
 border-left-color borderLeftColor Linker Rahmen
 border-left-style borderLeftStyle Linker Rahmen
 border-left-width borderLeftWidth Linker Rahmen
 border-right borderRight Rechter Rahmen
 border-right-color borderRightColor Rechter Rahmen
 border-right-style borderRightStyle Rechter Rahmen
 border-right-width borderRightWidth Rechter Rahmen
 border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
 border-top borderTop Oberer Rahmen
 border-top-color borderTopColor Oberer Rahmen
 border-top-style borderTopStyle Oberer Rahmen
 border-top-width borderTopWidth Oberer Rahmen
 border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
 font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 fontWeight Sets or retrieves the numeric weight of the font of the object.
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
 height height Objekthöhe ermitteln bzw. setzen
 layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen



layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
 margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
 margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
 margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
 margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
 max-height maxHeight Sets or retrieves the maximum height for an element.
 max-width maxWidth Sets or retrieves the maximum width for an element.
 min-height minHeight Sets or retrieves the minimum height for an element.
 min-width minWidth Sets or retrieves the minimum width for an element.
 padding padding Padding ermitteln bzw. setzen
 padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
 padding-left paddingLeft Padding-left ermitteln bzw. setzen
 padding-right paddingRight Padding-right ermitteln bzw. setzen
 padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
 page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
 page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
 pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 text-decoration-blink textBlinken ein aus
 text-decoration-line-through Text durchstreichen ein aus
 text-decoration-none Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 text-decoration-overline Linie über Text zeichnen ein aus
 text-decoration-underline Linie unter Text zeichnen ein aus
 text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 width width Breite eines Objektes ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.38. TD

Attribute

ABBR abbr Sets or retrieves abbreviated text for the object.
 ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
 ALIGN align Ausrichtung des Objektes beim Rendern
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 AXIS axis Sets or retrieves a comma-delimited list of conceptual categories associated with the object.
 BACKGROUND background Sets or retrieves the background picture tiled behind the text and graphics in the object.
BGCOLOR bgColor Diese Eigenschaft ist nicht mehr verwendbar !
 BORDERCOLOR borderColor Sets or retrieves the border color of the object.
 borderColorDark Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 borderColorLight Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 cellIndex Retrieves the position of the object in the cells collection of a row.
 CH ch Sets or retrieves a value that you can use to implement your own ch functionality for the object.
 CHOFF chOff Sets or retrieves a value that you can use to implement your own chOff functionality for the object.
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 COLSPAN colSpan Sets or retrieves the number columns in the table that the object should span.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 HEADERS headers Sets or retrieves a list of header cells that provide information for the object.



Height height Objekthöhe ermitteln bzw. setzen
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 NOWRAP noWrap automatischer Zeilenumbruch ein aus
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 ROWSPAN rowSpan Sets or retrieves how many rows in a table the cell should span.
 SCOPE scope Sets or retrieves the group of cells in a table to which the object's information applies.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermittel (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 VALIGN vAlign Sets or retrieves how text and other content are vertically aligned within the object that contains them.
 WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschneitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird



onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
onfocusin Fires for an element just prior to setting focus on that element.
onfocusout Fires for the current element with focus immediately after moving focus to another element.
onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
onmouseover Event erzeugt, sobald Maus das Objekt betritt
onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
onmousewheel Fires when the wheel button is rotated.
onmove Event erzeugt, wenn Objekt bewegt wird
onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
contains Ermitteln ob Objekt andere Objekte enthält
detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
dragDrop Drag-Event für das Objekt erzeugen
fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
getAdjacentText Benachbarten Textstring liefern
getAttribute Wert eines Attributes (einer Eigenschaft) liefern
getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
getExpression Ausdruck für eine Eigenschaft liefern
hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
insertAdjacentElement Nachbar-Objekt einfügen
insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
insertAdjacentText Text als Nachbar-Text einfügen
insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
mergeAttributes Alle Attribute in ein anderes Objekt kopieren
normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
releaseCapture Maus-Event-Überwachung zum Objekt entfernen
removeAttribute Attribute aus einem Objekt entfernen
removeAttributeNode Attribut aus Objekt entfernen
removeBehavior Behavior aus einem Objekt entfernen
removeChild Kindknoten aus einem Objekt entfernen
removeExpression Ausdruck aus einer Eigenschaft entfernen
removeNode Objektknoten aus Dokument-Hierarchie entfernen
replaceAdjacentText Benachbarten Text ersetzen
replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
replaceNode Knoten durch anderen Knoten ersetzen
scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
setActive Objekt aktiv setzen, aber nicht Focus geben
setAttribute Wert eines Attributes setzen
setAttributeNode Attribut-Knoten in Objekt einfügen
setCapture Maus-Event-Überwachung zum Objekt starten
setExpression Ausdruck setzen
swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen



Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.

:hover :hover Sets the style of an element when the user hovers the mouse pointer over it.

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen

border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen

border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen

border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen

border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen

border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen

border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen

border-left borderLeft Linker Rahmen

border-left-color borderLeftColor Linker Rahmen

border-left-style borderLeftStyle Linker Rahmen

border-left-width borderLeftWidth Linker Rahmen

border-right borderRight Rechter Rahmen

border-right-color borderRightColor Rechter Rahmen

border-right-style borderRightStyle Rechter Rahmen

border-right-width borderRightWidth Rechter Rahmen

border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen

border-top borderTop Oberer Rahmen

border-top-color borderTopColor Oberer Rahmen

border-top-style borderTopStyle Oberer Rahmen

border-top-width borderTopWidth Oberer Rahmen

border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen

clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf

clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen

color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)

cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen

direction direction Leserichtung ermitteln bzw. setzen

display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)

filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen

font font anhand Fonteigenschaftenliste ermitteln bzw. setzen

font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen

font-size fontSize Höhe des Fonts ermitteln bzw. setzen

font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen

font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)

fontWeight Sets or retrieves the numeric weight of the font of the object.

font-weight fontWeight Font-Wichtung ermitteln bzw. setzen

hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat

height height Objekthöhe ermitteln bzw. setzen

layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen

layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen

layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen

letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen

line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen

line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen

margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen

margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen

margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen

margin-right marginRight Dicke von right-margin ermitteln bzw. setzen

margin-top marginTop Höhe von top-margin ermitteln bzw. setzen

max-height maxHeight Sets or retrieves the maximum height for an element.

min-height minHeight Sets or retrieves the minimum height for an element.

padding padding Padding ermitteln bzw. setzen

padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen

padding-left paddingLeft Padding-left ermitteln bzw. setzen

padding-right paddingRight Padding-right ermitteln bzw. setzen

padding-top paddingTop Padding-Top ermitteln bzw. ersetzen

page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf

pixelBottom Position der unteren Kantes des Objektes ermitteln bzw. setzen

pixelHeight Höhe des Objektes ermitteln bzw. setzen

pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen

pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen



pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
 text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.39. TH

Attribute

ABBR abbr Sets or retrieves abbreviated text for the object.
 ACCESSKEY Kurztastenkombination ermitteln bzw. setzen
 ALIGN align Ausrichtung des Objektes beim Rendern
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 AXIS axis Sets or retrieves a comma-delimited list of conceptual categories associated with the object.
 BACKGROUND background Sets or retrieves the background picture tiled behind the text and graphics in the object.
 BGCOLOR bgColor Diese Eigenschaft ist nicht mehr verwendbar !
 BORDERCOLOR borderColor Sets or retrieves the border color of the object.
 borderColorDark Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 borderColorLight Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 cellIndex Retrieves the position of the object in the cells collection of a row.
 CH ch Sets or retrieves a value that you can use to implement your own ch functionality for the object.
 CHOFF chOff Sets or retrieves a value that you can use to implement your own chOff functionality for the object.
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 COLSPAN colSpan Sets or retrieves the number columns in the table that the object should span.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 HEADERS headers Sets or retrieves a list of header cells that provide information for the object.
 Height height Objekthöhe ermitteln bzw. setzen
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML HTML-Code zwischen den Begrenzer-Tags ermitteln bzw. setzen (inklusive .innerText)
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEditable Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 NOWRAP noWrap automatischer Zeilenumbruch ein aus



offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 ROWSPAN rowSpan Sets or retrieves how many rows in a table the cell should span.
 SCOPE scope Sets or retrieves the group of cells in a table to which the object's information applies.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermittelt (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 VALIGN vAlign Sets or retrieves how text and other content are vertically aligned within the object that contains them.
 WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresizing Event erzeugt, wenn die Änderung der Größe (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Größe (Dimension) des Objektes beginnt



onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern

getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

getExpression Ausdruck für eine Eigenschaft liefern

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

insertAdjacentHTML HTML-Code als Nachbar-Code einfügen

insertAdjacentText Text als Nachbar-Text einfügen

insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten

releaseCapture Maus-Event-Überwachung zum Objekt entfernen

removeAttribute Attribute aus einem Objekt entfernen

removeAttributeNode Attribut aus Objekt entfernen

removeBehavior Behavior aus einem Objekt entfernen

removeChild Kindknoten aus einem Objekt entfernen

removeExpression Ausdruck aus einer Eigenschaft entfernen

removeNode Objektknoten aus Dokument-Hierarchie entfernen

replaceAdjacentText Benachbarten Text ersetzen

replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen

replaceNode Knoten durch anderen Knoten ersetzen

scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)

setActive Objekt aktiv setzen, aber nicht Focus geben

setAttribute Wert eines Attributes setzen

setAttributeNode Attribut-Knoten in Objekt einfügen

setCapture Maus-Event-Überwachung zum Objekt starten

setExpression Ausdruck setzen

swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.

:hover :hover Sets the style of an element when the user hovers the mouse pointer over it.

background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen

background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen

background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)

background-image backgroundImage Hintergrundbild ermitteln bzw. setzen

background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen

behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen

border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen

border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen

border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen

border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen

border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen

border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen

border-left borderLeft Linker Rahmen

border-left-color borderLeftColor Linker Rahmen

border-left-style borderLeftStyle Linker Rahmen

border-left-width borderLeftWidth Linker Rahmen

border-right borderRight Rechter Rahmen

border-right-color borderRightColor Rechter Rahmen

border-right-style borderRightStyle Rechter Rahmen

border-right-width borderRightWidth Rechter Rahmen



border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
direction direction Leserichtung ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
fontWeight Sets or retrieves the numeric weight of the font of the object.
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-char layoutGridChar Gittergrösse für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for an element.
min-height minHeight Sets or retrieves the minimum height for an element.
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
textDecorationBlink Textblinken ein aus
textDecorationLineThrough Text durchstreichen ein aus
textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
textDecorationOverline Linie über Text zeichnen ein aus
textDecorationUnderline Linie unter Text zeichnen ein aus
text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.
visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
word-break wordBreak Zeilenumbruch in wörten bei mehrsprachigem Text ermitteln bzw. setzen



word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.40. TR

Attribute

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ALIGN align Ausrichtung des Objektes beim Rendern
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BGCOLOR bgColor Diese Eigenschaft ist nicht mehr verwendbar !
 BORDERCOLOR borderColor Sets or retrieves the border color of the object.
 borderColorDark Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 borderColorLight Sets or retrieves the color for one of the two colors used to draw the 3-D border of the object.
 canHaveChildren Status vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CH ch Sets or retrieves a value that you can use to implement your own ch functionality for the object.
 CHOFF chOff Sets or retrieves a value that you can use to implement your own chOff functionality for the object.
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 Height height Objekthöhe ermitteln bzw. setzen
 HIDEFOCUS hideFocus Status zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerHTML Retrieves the HTML between the start and end tags of the object.
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 readyState Retrieves a value that indicates the current state of the object.
 rowIndex Retrieves the position of the object in the rows collection for the table.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sectionRowIndex Retrieves the position of the object in the tBody, tHead, tFoot, or rows collection.
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 STYLE style inline-Style des Objektes setzen
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 VALIGN vAlign Sets or retrieves how text and other content are vertically aligned within the object that contains them.
 WIDTH width Sets or retrieves the width of the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird



onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird

onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird

onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird

onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält

onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird

onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)

onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)

oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)

oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt

oncopy Event erzeugt, wenn Objekt als selektierte Quelle in die Zwischenablage von Windows kopiert wird

oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird

ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde

ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird

ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)

ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde

ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist

ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf

ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert

ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird

onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet

onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren

onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird

onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird

onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird

onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert

onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)

onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren

onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen

onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt

onmouseout Event erzeugt, sobald die Maus das Objekt verlässt

onmouseover Event erzeugt, sobald Maus das Objekt betritt

onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt

onmousewheel Fires when the wheel button is rotated.

onmove Event erzeugt, wenn Objekt bewegt wird

onmoveend Event erzeugt, wenn Bewegung eines Objektes endet

onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt

onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden

onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet

onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt

onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren

ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

deleteCell Removes the specified cell (td) from the table row, as well as from the cells collection.

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern

getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

getExpression Ausdruck für eine Eigenschaft liefern



hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 insertCell Creates a new cell in the table row (tr), and adds the cell to the cells collection.
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Ort des Dynamic HTML (DHTML)-Behavior ermitteln bzw. setzen
 clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
 clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
 color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
 cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
 direction direction Leserichtung ermitteln bzw. setzen
 display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
 font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
 font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
 font-size fontSize Höhe des Fonts ermitteln bzw. setzen
 font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
 font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
 fontWeight Sets or retrieves the numeric weight of the font of the object.
 font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
 height height Objekthöhe ermitteln bzw. setzen
 layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-char layoutGridChar Gittergröße für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-line layoutGridLine Gitterwert für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layout ermitteln bzw. setzen
 layout-grid-type layoutGridType Gitter-Typ für Textzeichen-Layout ermitteln bzw. setzen
 letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
 line-break lineBreak Zeilenumbruchregel für Japanisch ermitteln bzw. setzen
 line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
 max-height maxHeight Sets or retrieves the maximum height for an element.
 min-height minHeight Sets or retrieves the minimum height for an element.
 page-break-after pageBreakAfter Status ermitteln bzw. setzen, ob Seitenumbruch nach einem Objekt erfolgen darf
 page-break-before pageBreakBefore Status ermitteln bzw. setzen, ob Seitenumbruch vor einem Objekt erfolgen darf
 pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen
 pixelHeight Höhe des Objektes ermitteln bzw. setzen
 pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
 pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen



text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-indent textIndent Ausrichtung der ersten Textlinie im Objekt ermitteln bzw. setzen
 text-justify textJustify Blockausrichtung von Text im Objekt ermitteln bzw. setzen
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 vertical-align verticalAlign Sets or retrieves the vertical alignment of the object.
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 word-break wordBreak Zeilenumbruch in wörtern bei mehrsprachigem Text ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.B. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.41. TEXTAREA

Attribute

Attribute Property Description

ACCESSKEY Kurzstastenkombination ermitteln bzw. setzen
 ATOMICSELECTION Objekt und Kontext als Ganzheit selektierbar setzen
 BEGIN begin Wartezeit vor Play-Start auf der Timeline ermitteln bzw. setzen
 canHaveChildren Stauts vom Objekt ermitteln, ob Objekt Kinder haben darf
 canHaveHTML Status vom Objekt ermitteln, ob Objekt HTML Markups enthalten darf
 CLASS className Klassenname des Objektes ermitteln bzw. setzen
 clientHeight Objekthöhe (inklusive padding, aber ohne margin und border und scroll bar)
 clientLeft Abstand des Objektes zwischen .offsetLeft und linker Kante des Clientfensters
 clientTop Abstand des Objektes zwischen .offsetTop und oberer Kante des Clientfensters
 clientWidth Objektbreite (inklusive padding, aber ohne margin und border und scroll bar)
 COLS cols Sets or retrieves the width of the object.
 CONTENTEDITABLE contentEditable User-Editierbarkeit des Objektes im Kontext ermitteln bzw. setzen
 DATAFLD dataFld Feld der Datenquelle ermitteln bzw. setzen, das in das Objekt eingebunden werden soll bzw. eingebunden ist
 DATASRC dataSrc Datenquelle für Dateneinbindung in ein Objekt ermitteln bzw. setzen
 defaultValue Sets or retrieves the initial contents of the object.
 DIR dir Lesereichtung des Objektes ermitteln bzw. setzen
 disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)
 DISABLED disabled Verfügbarkeitsstatus des Objektes ermitteln bzw. setzen
 END end Endezeit oder Wiederholungsdauer vom Play des Objektes auf Timeline ermitteln bzw. setzen
 firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt
 form Retrieves a reference to the form that the object is embedded in.
 hasMedia Status (booealan) zum Objekt ermitteln, ob Objekt ein HTML+TIME media element ist
 HIDEFOCUS hideFocus Stauts zum sichtbaren Objekt ermitteln bzw. setzen, ob Objekt den Focus hat
 ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())
 innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen
 isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist
 isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann
 isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist
 isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist
 LANG lang zu benutzende Sprache ermitteln bzw. setzen
 LANGUAGE language Scriptsprache ermitteln bzw. setzen
 lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt
 NAME name Bezeichner als Name vom Objekt ermitteln bzw. setzen (ist nicht das ID des Objektes, dient der Gruppierung)
 nextSibling Referenz auf das nächste Kind der Eltern vom Objekt
 nodeName Name des Teiltypen des Knoten ermitteln
 nodeType Type des Knoten, auf den zugegriffen wird, ermitteln
 nodeValue Wert eines Knoten ermitteln bzw. setzen
 offsetHeight Höhenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetLeft Left des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetParent Zeiger auf Elternobjekt ermitteln für .offsetTop und .offsetLeft
 offsetTop Top des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 offsetWidth Breitenabstand des Objektes relativ zu den Eltern laut .offsetParent ermitteln
 onOffBehavior Status vom Objekt ermitteln, ob eine Microsoft Direct Animation mit dem Objekt abläuft
 outerHTML HTML-Kontent um das Objekt ermitteln bzw. setzen
 outerText Text-Kontent um das Objekt ermitteln bzw. setzen
 ownerDocument Retrieves the document object associated with the node.
 parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln
 parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln
 parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf
 previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln
 READONLY readOnly Sets or retrieves whether the content of the object is read-only.



readyState Status des Objektes ermitteln
 recordNumber Nummer des Satzes aus dem Daten-Set des Objektes ermitteln
 ROWS rows Sets or retrieves the number of horizontal rows contained in the object.
 scopeName Namensbereich des Objektes ermitteln
 scrollHeight Scrollhöhe vom Objekt ermitteln
 scrollLeft Abstand linke Kante Objekt zur linken Kante Elternobjekt ermitteln bzw. setzen
 scrollTop Abstand obere Kante Objekt zur oberen Kante Elternobjekt ermitteln bzw. setzen
 scrollWidth Scrollbreite vom Objekt ermitteln
 sourceIndex Index des Objektes in der Collection document.all ermitteln
 status Sets or retrieves the value indicating whether the control is selected.
 STYLE style inline-Style des Objektes setzen
 SYNCMASTER syncMaster Status ermitteln, ob Time-Container mit dem Objekt im Play synchronisiert werden muss
 systemBitrate verfügbare Bandbreite in Bits pro Sekunde ermitteln
 systemCaptions Status ermitteln, ob Text-Äquivalent zu einem Audio angezeigt wird
 systemLanguage Status ermitteln, ob Sprache selektiert wurde in der UserEinstellung
 systemOverdubOrSubtitle Status ermitteln, ob Subtitel renderbar sind
 TABINDEX tabIndex Index des Objektes in der Tab-Folge ermitteln bzw. setzen (Tabfolge bei Tab-Tastendruckfolge)
 tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)
 tagUri Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen
 TIMECONTAINER timeContainer Typ der Timeline zum Objekt ermitteln bzw. setzen
 TITLE title Tooltipp (Textblase bei mouseover) zum Objekt ermitteln bzw. setzen (nicht Titel des Dokumentes)
 type Retrieves the type of control.
 uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)
 UNSELECTABLE Status der Nicht-Selektierbarkeit eines Objektes ermitteln bzw. setzen
 value Retrieves or sets the text in the entry field of the textArea element.
 WRAP wrap Sets or retrieves how to handle wordwrapping in the object.

Events

onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterupdate Event erzeugt direkt nach erfolgreichem Update der Daten
 onbeforeactivate Fires immediately before the object is set as the active element.
 onbeforecopy Event erzeugt direkt bevor das Objekt in die Zwischenablage von Windows kopiert wird
 onbeforecut Event erzeugt direkt bevor das selektierte Objekt aus dem Dokument ausgeschnitten wird
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeeditfocus Event erzeugt direkt bevor Objekt den Editier-Fokus erhält
 onbeforepaste Event erzeugt direkt bevor Objekt aus Windows Zwischenablage in das Ziel kopiert wird
 onbeforeupdate Event erzeugt direkt vor dem Update der Daten
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 onchange Fires when the contents of the object or selection have changed.
 onclick Event erzeugt, wenn das Objekt mit der linken Maustaste geklickt wird (Maus steht bereits über dem Objekt)
 oncontextmenu Event erzeugt, wenn das Objekt mit der rechten Maustaste geklickt wird, um ein Kontext-Menü zu aktivieren (Maus steht bereits über dem Objekt)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 oncut Event erzeugt, wenn Objekt als selektiertes Objekt aus dem Dokument ausgeschnitten und in die Zwischenablage von Windows kopiert wird
 ondblclick Event erzeugt, wenn Objekt mit Maus doppelgeklickt wurde
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 ondrag Event erzeugt, wenn selektiertes Objekt verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragend Event erzeugt, wenn selektiertes Objekt das letzte Mal verschoben wird (nicht ausschneiden, nicht kopieren etc.)
 ondragenter Event erzeugt, wenn selektiertes Objekt nach Verschiebung in das Ziel-Objekt abgelegt wurde
 ondragleave Event erzeugt, wenn Drag & Drop nicht möglich ist
 ondragover Event erzeugt, wenn selektiertes Objekt sich über eine Ziel befindet, in das abgelegt werden darf
 ondragstart Event erzeugt, wenn User eine Textselektion auf das Objekt aktiviert
 ondrop Event erzeugt, wenn selektiertes Objekt nach der Verschiebung im Ziel gerade abgelegt wird
 onerrorupdate Event erzeugt, wenn Fehler beim Update der Daten erkannt wurde
 onfilterchange Event erzeugt, wenn ein Filter den Status ändert oder eine Transition endet
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onfocusin Fires for an element just prior to setting focus on that element.
 onfocusout Fires for the current element with focus immediately after moving focus to another element.
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onkeydown Event erzeugt, sobald eine Taste herunter gedrückt wird
 onkeypress Event erzeugt, sobald eine alpha-numerische Taste herunter gedrückt wird
 onkeyup Event erzeugt, sobald eine gedrückte Taste losgelassen wird
 onlosecapture Event erzeugt, wenn Objekt die Maus-Event-Überwachung verliert
 onmousedown Event erzeugt, wenn irgendeine Maustaste über dem Objekt gedrückt wird (Maus steht über dem Objekt)
 onmouseenter Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu überfahren
 onmouseleave Event erzeugt, wenn Mauszeiger das Objekt gerade beginnt zu verlassen
 onmousemove Event erzeugt, sobald Maus bewegt wird über einem Objekt
 onmouseout Event erzeugt, sobald die Maus das Objekt verlässt
 onmouseover Event erzeugt, sobald Maus das Objekt betritt
 onmouseup Event erzeugt, wenn gedrückte Maustaste losgelassen wird über einem Objekt
 onmousewheel Fires when the wheel button is rotated.
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt



onpaste Event erzeugt, wenn Daten aus der Zwischenablage von Windows in das Ziel abgelegt werden
 onpropertychange Event erzeugt, wenn sich eine Eigenschaft oder ein Attribut-Wert des Objektes ändert
 onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizeend Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onselect Event erzeugt wenn die Selektion des Objektes geändert wird
 onselectstart Event erzeugt, wenn begonnen wird, das Objekt zu selektieren
 ontimeerror Event erzeugt, wenn Zeitfehler wegen falschem Wert für eine Eigenschaft eintritt

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen
 appendChild Dem Objekt ein bereits erzeugtes Kind hinzufügen durch Anhängen hinter das bisher letzte Kind
 applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen
 click Für das Objekt das Event onclick erzeugen (ansonsten .fireEvent() verwenden)
 cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren
 componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten
 contains Ermitteln ob Objekt andere Objekte enthält
 createTextRange TextRange-Objekt erzeugen
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 doScroll Erzeugt Klick auf die Scrollbar des Objektes
 dragDrop Drag-Event für das Objekt erzeugen
 fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 getAdjacentText Benachbarten Textstring liefern
 getAttribute Wert eines Attributes (einer Eigenschaft) liefern
 getAttributeNode Referenz auf Attribute-Objekt nach Bezeichner der Eigenschaft liefern
 getBoundingClientRect Grenzen von TextRange-Objekten als Objekt liefern
 getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)
 getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln
 getExpression Ausdruck für eine Eigenschaft liefern
 hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht
 insertAdjacentElement Nachbar-Objekt einfügen
 insertAdjacentHTML HTML-Code als Nachbar-Code einfügen
 insertAdjacentText Text als Nachbar-Text einfügen
 insertBefore Objekt als Kind-Knoten in die Dokument-Hierarchie einfügen
 mergeAttributes Alle Attribute in ein anderes Objekt kopieren
 normalize Benachbarte Textknoten mischen um ein normalisiertes DOM zu erhalten
 releaseCapture Maus-Event-Überwachung zum Objekt entfernen
 removeAttribute Attribute aus einem Objekt entfernen
 removeAttributeNode Attribut aus Objekt entfernen
 removeBehavior Behavior aus einem Objekt entfernen
 removeChild Kindknoten aus einem Objekt entfernen
 removeExpression Ausdruck aus einer Eigenschaft entfernen
 removeNode Objektknoten aus Dokument-Hierarchie entfernen
 replaceAdjacentText Benachbarten Text ersetzen
 replaceChild Kind-Objekt durch anderes Kind-Objekt ersetzen
 replaceNode Knoten durch anderen Knoten ersetzen
 scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)
 select Highlights the input area of a form element.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setAttribute Wert eines Attributes setzen
 setAttributeNode Attribut-Knoten in Objekt einfügen
 setCapture Maus-Event-Überwachung zum Objekt starten
 setExpression Ausdruck setzen
 swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

:first-child :first-child Applies one or more styles to any element that is the first child of its parent.
 :hover :hover Sets the style of an element when the user hovers the mouse pointer over it.
 ACCELERATOR accelerator Tastaturkürzel setzen
 background background Hintergrund-Eigenschaften aus 5 Komponenten ermitteln bzw. setzen
 background-attachment backgroundAttachment Art des Rederns des Hintergrundbildes ermitteln bzw. setzen
 background-color backgroundColor Hintergrundfarbe ermitteln bzw. setzen (ist nicht Textfarbe)
 background-image backgroundImage Hintergrundbild ermitteln bzw. setzen
 background-position backgroundPosition Position des Hintergrundes ermitteln bzw. setzen
 background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen
 background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen
 background-repeat backgroundRepeat Wiederholung des Hintergrundbildes beim Rendern ermitteln bzw. setzen
 behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen
 border border Layout des Rahmens um das Objekt an den Objektgrenzen ermitteln bzw. setzen



border-bottom borderBottom unteren Rahmen ermitteln bzw. setzen
border-bottom-color borderBottomColor Farbe des unteren Rahmen ermitteln bzw. setzen
border-bottom-style borderBottomStyle Layout des unteren Rahmen ermitteln bzw. setzen
border-bottom-width borderBottomWidth Dicke des unteren Rahmen ermitteln bzw. ersetzen
border-color borderColor Rahmenfarbe ermitteln bzw. ersetzen
border-left borderLeft Linker Rahmen
border-left-color borderLeftColor Linker Rahmen
border-left-style borderLeftStyle Linker Rahmen
border-left-width borderLeftWidth Linker Rahmen
border-right borderRight Rechter Rahmen
border-right-color borderRightColor Rechter Rahmen
border-right-style borderRightStyle Rechter Rahmen
border-right-width borderRightWidth Rechter Rahmen
border-style borderStyle Layout der Rahmen links rechts oben unten ermitteln bzw. setzen
border-top borderTop Oberer Rahmen
border-top-color borderTopColor Oberer Rahmen
border-top-style borderTopStyle Oberer Rahmen
border-top-width borderTopWidth Oberer Rahmen
border-width borderWidth Rahmenbreite links recht oben unten ermitteln bzw. setzen
bottom bottom Rahmenposition ermitteln bzw. setzen relativ zum Nachbarobjekt laut Dokument-Hierarchie
clear clear Status ermitteln bzw. setzen, ob Objekt durch andere Objekte umflossen werden darf
clip clip Sichtbaren Bereich eines Objektes ermitteln bzw. setzen
color color Textfarbe ermitteln bzw. setzen (Text erscheint auf Hintergrund)
cursor cursor Maus-Cursor-Layout ermitteln bzw. setzen
direction direction Leserichtung ermitteln bzw. setzen
display display Status ermitteln bzw. setzen, ob Objekt gerendert werden darf (siehe auch visibility)
filter filter Collection der zum Objekt verfügbaren Filter ermitteln bzw. setzen
font font font anhand Fonteigenschaftenliste ermitteln bzw. setzen
font-family fontFamily Bezeichner des Fontfamilie ermitteln bzw. setzen
font-size fontSize Höhe des Fonts ermitteln bzw. setzen
font-style fontStyle Font-Stil italic normal oblique ermitteln bzw. setzen
font-variant fontVariant Font-Variante ermitteln bzw. setzen (small capital letters)
fontWeight Sets or retrieves the numeric weight of the font of the object.
font-weight fontWeight Font-Wichtung ermitteln bzw. setzen
hasLayout Status ermitteln bzw. setzen, ob Objekt Layout hat
height height Objekthöhe ermitteln bzw. setzen
ime-mode imeMode Sets or retrieves the state of an Input Method Editor (IME).
layout-flow layoutFlow Flussrichtung und Flussart des Kontent des Objektes ermitteln bzw. setzen
layout-grid layoutGrid Gitter-Eigenschaften für Textzeichen-Layout ermitteln bzw. setzen
layout-grid-mode layoutGridMode Dimension des Gitters (1 oder 2 Dimensionen) für Textzeichen-Layozt ermitteln bzw. setzen
left left X-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
letter-spacing letterSpacing Buchstabenabstand ermitteln bzw. setzen
line-height lineHeight Zeilenabstand zwischen Linien ermitteln bzw. setzen
margin margin Breite von top-, right-, bottom-, and left-margins ermitteln bzw. setzen
margin-bottom marginBottom Höhe von bottom-margin ermitteln bzw. setzen
margin-left marginLeft Dicke von left-margin ermitteln bzw. setzen
margin-right marginRight Dicke von right-margin ermitteln bzw. setzen
margin-top marginTop Höhe von top-margin ermitteln bzw. setzen
max-height maxHeight Sets or retrieves the maximum height for displayable block level elements.
max-width maxWidth Sets or retrieves the maximum width for displayable block level elements.
min-height minHeight Sets or retrieves the minimum height for displayable block level elements.
min-width minWidth Sets or retrieves the minimum width for displayable block level element.
overflow overflow Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension ausdehnen darf
overflow-x overflowX Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf X-Achse ausdehnen darf
overflow-y overflowY Status ermitteln bzw. setzen, ob Content des Objektes die Objekt-Dimension auf Y-Achse ausdehnen darf
padding padding Padding ermitteln bzw. setzen
padding-bottom paddingBottom Padding-bottom ermitteln bzw. setzen
padding-left paddingLeft Padding-left ermitteln bzw. setzen
padding-right paddingRight Padding-right ermitteln bzw. setzen
padding-top paddingTop Padding-Top ermitteln bzw. ersetzen
pixelBottom Postion der unteren Kantes des Objektes ermitteln bzw. setzen
pixelHeight Höhe des Objektes ermitteln bzw. setzen
pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen
pixelRight Position der rechten Kantes des Objektes ermitteln bzw. setzen
pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
pixelWidth Breite des Objektes ermitteln bzw. setzen
posBottom Bottom-Position ermitteln bzw. setzen
posHeight Höhe ermitteln bzw. setzen
position position Art der Positionierung (relative, absolute) ermitteln bzw. setzen
posLeft Left ermitteln bzw. setzen
posRight Right ermitteln bzw. setzen
posTop Top ermitteln bzw. setzen
posWidth Breite ermitteln bzw. setzen
right right Right ermitteln bzw. setzen



scrollbar-3dlight-color scrollbar3dLightColor Farbe der oberen und linken Kante von Scroobar und Scroll-Pfeile ermitteln bzw. setzen
 scrollbar-arrow-color scrollbarArrowColor Farbe der Scrol-Pfeile ermitteln bzw. setzen
 scrollbar-base-color scrollbarBaseColor Farbe der Hauptelement der Scrollbar, Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-darkshadow-color scrollbarDarkShadowColor Schattenfarbe der Scrollbar ermitteln bzw. setzen
 scrollbar-face-color scrollbarFaceColor Farbe der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-highlight-color scrollbarHighlightColor Farbe der oberen und linken Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-shadow-color scrollbarShadowColor Farbe der unteren und der rechten Kante der Scrollbox und Scrollpfeile ermitteln bzw. setzen
 scrollbar-track-color scrollbarTrackColor Farbe der Trackelemente der Scrollbar ermitteln bzw. setzen
 float styleFloat Seite des Objektes ermitteln bzw. setzen, ander der Fluss erfolgt
 text-align textAlign Textausrichtung ermitteln bzw. setzen (linksbündig, rechtsbündig, zentriert, block)
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-decoration textDecoration Textlayout blink, line-through, overline, underline (Dekoration) ermitteln bzw. setzen
 textDecorationBlink Textblinken ein aus
 textDecorationLineThrough Text durchstreichen ein aus
 textDecorationNone Verfügbarkeit von Text-Eigenschaft Dekoration ein aus
 textDecorationOverline Linie über Text zeichnen ein aus
 textDecorationUnderline Linie unter Text zeichnen ein aus
 text-overflow textOverflow Sets or retrieves a value that indicates whether to render ellipses(...) to indicate text overflow.
 text-transform textTransform Art des Rendern von Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen
 top top Y-Ordinate der Position der linken oberen Ecke des Objektes ermitteln bzw. setzen (beachte position)
 unicode-bidi unicodeBidi Stufe der eingebetteten bidirektionalen Algorithmen ermitteln bzw. setzen
 visibility visibility Sichtbarkeit eines Objektes ermitteln bzw. setzen (Layout der Objektumgebung wird z.T. automatisch angepasst)
 width width Breite eines Objektes ermitteln bzw. setzen
 word-spacing wordSpacing Zusatzleerzeichen zwischen Wörtern ermitteln bzw. setzen
 word-wrap wordWrap automatischer Zeilenumbruch bei Wörtern ein aus
 writing-mode writingMode Richtung und Fluss des Content vom Objekt ermitteln bzw. setzen
 z-index zIndex Position in der Renderfolge von z.b. sich überlagernden Objekten ("Layer" des Internet Explorers)
 zoom zoom Vergrößerung oder Verkleinerung der gerenderten (sichtbaren) Dimension eines Objektes

4.3.2.2.12.9.42. TextRange

Textrange liest den Quelltext des HTML-Dokumentes aus und weist das Leseregebnis .text zu
 wird in der aktuellen Dimension ausgelesen:

.text muss also NICHT den Stand haben wie zum Zeitpunkt createTextRange()
 createTextRange() instanziert NUR das Textrange-Objekt und belegt nicht .text statisch !!!
 per style.visibility='hidden' lässt sich kein Objekt mit Text z.B. Textarea ausblenden
 Beispiel:

```

<SCRIPT LANGUAGE="JScript">
var ZeigerFeldDerButtons = document.all.tags("BUTTON");    // Zeiger aller BUTTON-Objekte als Collection ermitteln
if (ZeigerFeldDerButtons!=null)
{var TextBereich = ZeigerFeldDerButtons [0].createTextRange(); // Erstes gefundene Button-Objekt mit Textrange versehen
                                // Der Textbereich des Buttons wird also eingestellt
    if (TextBereich != null) {TextBereich.text = "Clicked";}    // Textbereich des Button neu setzten
}
</SCRIPT>
  
```

Beispiel: für Suchen in einem Dokument:

```

<script language="JavaScript">
var NS4=(document.layers);
var IE4=(document.all);
var win=window;
var n=0;

function findInPage(str)
{
  var txt,i,found;
  if(str=="")
  return false;
  if(NS4)
  {
    if(!win.find(str))
    while(win.find(str,false,true))
    n++;
    else
    n++;
    if(n==0)
    alert("Nichts gefunden.");
  }
  if(IE4)
  {
    txt=win.document.body.createTextRange();
    for(i=0;i<=n&&(found=txt.findText(str))!=false;i++)
    {
  
```



```

    txt.moveStart("character",1);
    txt.moveEnd("textedit");
  }
  if(found)
  {
    txt.moveStart("character",-1);
    txt.findText(str);
    txt.select();
    txt.scrollIntoView();
    n++;
  }
  else
  {
    if(n>0)
    {
      n=0;
      findInPage(str);
    }
    else
      alert("Nichts gefunden.");
  }
}
return false;
}
</script>
<form name="search" onSubmit="return findInPage(this.string.value);">
<font size=3><input name="string" type="text" size=15 onChange="n = 0;"></font>
<input type="submit" value="Suchen">
</form>

```

Attribute

boundingHeight Retrieves the height of the rectangle that bounds the TextRange object.

boundingLeft Retrieves the distance between the left edge of the rectangle that bounds the TextRange object and the left side of the object that contains the TextRange.

boundingTop Retrieves the distance between the top edge of the rectangle that bounds the TextRange object and the top side of the object that contains the TextRange.

boundingWidth Retrieves the width of the rectangle that bounds the TextRange object.

htmlText Retrieves the HTML source as a valid HTML fragment.

offsetLeft Retrieves the calculated left position of the object relative to the layout or coordinate parent, as specified by the offsetParent property.

offsetTop Retrieves the calculated top position of the object relative to the layout or coordinate parent, as specified by the offsetParent property.

text Sets or retrieves the text contained within the range.

Events

keine

Methoden

collapse Moves the insertion point to the beginning or end of the current range.

compareEndpoints Compares an end point of a TextRange object with an end point of another range.

duplicate Returns a duplicate of the TextRange.

execCommand Executes a command on the current document, current selection, or the given range.

expand Expands the range so that partial units are completely contained.

findText Searches for text in the document and positions the start and end points of the range to encompass the search string.

getBookmark Retrieves a bookmark (opaque string) that can be used with moveToBookmark to return to the same range.

getClientRect Grenzen von TextRange-Objekten als Objekt liefern

getClientRects Collection von Rechtecken zum Layout des Kontens vom Objekt im Client liefern (Jedes Rechteck ist eine Zeile)

inRange Returns a value indicating whether one range is contained within another.

isEqual Returns a value indicating whether the specified range is equal to the current range.

move Collapses the given text range and moves the empty range by the given number of units.

moveEnd Changes the end position of the range.

moveStart Changes the start position of the range.

moveToBookmark Moves to a bookmark.

moveToElementText Moves the text range so that the start and end positions of the range encompass the text in the given element.

moveToPoint Moves the start and end positions of the text range to the given point.

parentElement Retrieves the parent element for the given text range.

pasteHTML Pastes HTML text into the given text range, replacing any previous text and HTML elements in the range.

queryCommandEnabled Returns a Boolean value that indicates whether a specified command can be successfully executed using execCommand, given the current state of the document.

queryCommandIndeterm Returns a Boolean value that indicates whether the specified command is in the indeterminate state.

queryCommandState Returns a Boolean value that indicates the current state of the command.

queryCommandSupported Returns a Boolean value that indicates whether the current command is supported on the current range.

queryCommandValue Returns the current value of the document, range, or current selection for the given command.

scrollIntoView Objekt in einen View scrollen (entweder an obere oder untere Kante)

select Makes the selection equal to the current object.

setEndPoint Sets the endpoint of one range based on the endpoint of another range.

Styles

keine

4.3.2.2.12.9.43. TITLE im HEAD

Enthält den Titel des Dokumentes

document.title kann nicht durch (Folge von) Leerzeichen gelöscht werden: Es müssen Zeichen ungleich Leerzeichen enthalten sein !

läuft ein Dokument im Frameset, so

ist der sichtbare Titel im Fensterkopf DER vom FRAMESET und nicht vom Frame.

muss parent.document.title verwendet werden, um von Frame aus den sichtbaren Titel vom FRAMESET manipulieren zu können

Attribute

canHaveHTML Sets or retrieves the value indicating whether the object can contain rich HTML markup.

disabled Status ermitteln bzw. setzen, ob User mit Objekt agieren kann (ist nicht visibility)

firstChild Referenz auf das 1. Kind in der Collection childNodes vom Objekt

ID id Bezeichner als String eines Objektes ermitteln bzw. setzen (ist nicht NAME des Objektes, kann als Zeiger auf das Objekt verwendet werden, nicht kodieren im HTML-Code für createElement())

innerHTML Retrieves the HTML between the start and end tags of the object.

innerText Text, der nicht HTML-Code ist, zwischen den Begrenzer-Tags ermitteln bzw. setzen

isContentEditable Status vom Objekt ermitteln bzw. setzen, ob Objekt im Kontext editierbar ist

isDisabled Status vom Objekt ermitteln, ob User mit Objekt interagieren kann

isMultiLine Status vom Objekt ermitteln, ob Kontext des Objektes ein- oder mehrzeilig ist

isTextEdit Status ermitteln, ob ein TextRange im Objekt erzeugbar ist

LANG lang zu benutzende Sprache ermitteln bzw. setzen

lastChild Referenz auf das letzte Kind in der Collection childNodes vom Objekt

nextSibling Referenz auf das nächste Kind der Eltern vom Objekt

nodeName Name des Teiltypen des Knoten ermitteln

nodeType Type des Knoten, auf den zugegriffen wird, ermitteln

nodeValue Wert eines Knoten ermitteln bzw. setzen

ownerDocument Referenz auf Dokument ermitteln bzw. setzen, das mit dem Knoten verbunden ist

parentElement Referenz auf das Elternobjekt in der Objekthierarchie ermitteln

parentNode Referenzknoten des Elternobjektes in der Objekthierarchie ermitteln

parentTextEdit Status vom Objekt ermitteln, ob Elternobjekt einen TextRange aus dem Kind erzeugen darf

previousSibling Referenz auf das vorhergehende Kind im Elternobjekt ermitteln

readyState Status des Objektes ermitteln

scopeName Namensbereich des Objektes ermitteln

sourceIndex Index des Objektes in der Collection document.all ermitteln

tagName Tag-Bezeichner des Objektes ermitteln (keine spitzen Klammern)

tagUrn Uniform Resource Name (URN) laut Namensbereich-Deklaration ermitteln bzw. setzen

text Retrieves or sets the text of the object as a string.

uniqueID vom Browser intern erzeugt ID des Objektes ermitteln (kann als Zeiger verwendet werden, ist nicht ID-Attribut-Wert)

Events

onlayoutcomplete Event erzeugt wenn Druckvorschau oder Druck das LayoutRect-Objekt gefüllt haben

onreadystatechange Event erzeugt, wenn der Status eines Objektes sich ändert

Methoden

addBehavior Dem Objekt ein Behavior hinzufügen

applyElement Das Objekt als Eltern-oder Kind-Objekt für ein anderes Objekt setzen (Eltern so anderes Objekt wird 1. Kind, Kind so anderes Objekt erhält Kind eingefügt)

attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren

clearAttributes Aus dem Objekt ALLE Attribute (Eigenschaften) entfernen

cloneNode Referenz auf Knoten aus der Dokument-Hierarchie kopieren

componentFromPoint Komponenten liefern je nach Wert an bestimmten Koordinaten

contains Ermitteln ob Objekt andere Objekte enthält

detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen

dragDrop Drag-Event für das Objekt erzeugen

fireEvent Für das Objekt ein Event auslösen (fireEvent() nur innerhalb Eventhandler, cancelBubble wird false)

getAdjacentText Benachbarten Textstring liefern

getAttribute Wert eines Attributes (einer Eigenschaft) liefern

getAttributeNode Attribute-Objekt zu einem Eigenschaftsbezeichner ermitteln

getElementsByTagName Zeigern auf Objekte im Objekt anhand des Tagnamens ermitteln

hasChildNodes Ermitteln ob Objekt Kinder hat oder nicht

insertAdjacentElement Nachbar-Objekt einfügen

mergeAttributes Alle Attribute in ein anderes Objekt kopieren

normalize Benachbarte Textknoten mischen, um damit eine normale DOM-Struktur zu erhalten

removeAttribute Attribute aus einem Objekt entfernen

removeAttributeNode Attribut-Objekt aus einem Objekt entfernen

removeBehavior Behavior aus einem Objekt entfernen

replaceAdjacentText Benachbarten Text ersetzen

setAttribute Wert eines Attributes setzen

setAttributeNode Attribut-Objekt-Knoten einfügen

swapNode Position von 2 Objekten in der Dokument-Hierarchie tauschen

Styles

background-position-x backgroundPositionX Hintergrundposition X-Ordinate ermitteln bzw. setzen

background-position-y backgroundPositionY Hintergrund-Position Y-Ordinate ermitteln bzw. setzen

behavior behavior Location der Einführung eines DHTML-Behavior ermitteln bzw. setzen

pixelBottom Position der unteren Kante des Objektes ermitteln bzw. setzen

pixelHeight Höhe des Objektes ermitteln bzw. setzen

pixelLeft Position der linken Kante des Objektes ermitteln bzw. setzen



pixelRight Position der rechten Kante des Objektes ermitteln bzw. setzen
 pixelTop Position der oberen Kante des Objektes ermitteln bzw. setzen
 pixelWidth Breite des Objektes ermitteln bzw. setzen
 posBottom Bottom-Position ermitteln bzw. setzen
 posHeight Höhe ermitteln bzw. setzen
 posLeft Left ermitteln bzw. setzen
 posRight Right ermitteln bzw. setzen
 posTop Top ermitteln bzw. setzen
 posWidth Breite ermitteln bzw. setzen
 text-autospace textAutospace Autabstandsbildung für Text ermitteln bzw. setzen
 text-underline-position textUnderlinePosition Position der Unterstrichlinie ermitteln bzw. setzen

4.3.2.2.12.9.44. userProfile

Verwalten von Nutzerdaten im Datenbereich des IE

Beispiel:

```

// Lese-Zugriff in den Jobstapel einfügen
navigator.userProfile.addReadRequest("vcard.displayname");
navigator.userProfile.addReadRequest("vcard.gender");

// Job-Stapel abarbeiten also alle Zugriffe starten
navigator.userProfile.doReadRequest(usage-code, "Acme Corporation");

// Informationen auslesen
name = navigator.userProfile.getAttribute("vcard.displayname");
gender = navigator.userProfile.getAttribute("vcard.gender");

// Job-Stapel löschen
navigator.userProfile.clearRequest();
  
```

Es gibt noch andere Varianten der Verwaltung von Nutzerdaten (die der User ebenfalls nicht zwingend bemerken muss):

```

Cookies
behavior.url(#default#userData)
  
```

Attribute

keine

Events

keine

Methoden

addReadRequest Adds an entry to the queue for read requests.
 clearRequest Clears all requests in the read-requests queue to prepare for new profile-information requests.
 doReadRequest Performs all requests located in the read-requests queue.
 getAttribute Returns the value of the named attribute from the userProfile object.
 setAttribute Sets the value of the specified attribute.

Styles

keine

4.3.2.2.12.9.45. window

window ist Elternobjekt diverser Objekte wie document oder event

Apache-Server behandelt Dokument auf lokal Host (127.0.0.1), das per window.open() erzeugt wurde, nicht zwingend wie es im Script kodiert wurde.

```

wenn im open "status=no" angegeben wurden, dann kann Statuszeile trotzdem erscheinen
wenn im open "fullscreen=yes" angegeben wurden, dann erscheint eventuell kein fullscreen (Kiosk-Modus) sondern z.B. die
    Statuszeile
  
```

Es kann sein, dass der Firewall und/oder der IE in den Einstellungen einen Fullscreen verbieten.

Das Layout eines Window ändert sich mit Browserversion und kann inkompatibel sein (z.B. zusätzliche Statuszeile beim IE 7, die es beim IE 6 nicht gibt)

Kiosk-modus per window.open(.... 'fullscreen=yes')

```

impliziert automatisch 'channelmode=no,directories=no,location=no,menubar=no,status=no,titlebar=no,toolbar=no'
  
```

egal ob für diese Parameter 'no' oder 'yes' kodiert, wobei 'no' Standard ist

window.status und HREF-Attribut

```

Der IE erzeugt bei mouseover über <A> mit HREF z.B. auf Datei eine Änderung von windows.status
    mit mouseover wird HREF-Wert angezeigt
    mit mouseout wird window.status auf den Wert VOR mouseover gesetzt
    bei rechten Maustasteklick die Anzeige des HREF-Wertes solange, wie rechte Maustaste gedrückt bleibt.
  
```

```

Abhilfe dafür schafft return true; für
    onmouseover
  
```



onfocus da oncontextmenu nicht die Anzeige des HREF per rechter Maus unterdrückt

Beispiel:

```
<a href="link.htm" onmouseover="window.status='Text mit Mouseover';return true"
    onfocus="window.status='Text mit Mouseover';return true"
    onmouseout="window.status='Standardtext vor mouseover';return true">
das ist der link-Text
</a>
```

oncontextmenu und HREF-Attribut

Das Sperren von oncontextmenu unterdrückt nicht die Anzeige des HREF-Attributwertes: Es muss onfocus-Event behandelt werden.

Frameset im IE 6

Ohne FRAMSET gilt:

sämtliche per open erzeugte Fenster werden weder registriert per Colletion window.frames noch document.all
Es gibt keine Collection der Fenster
Die Fensterhierarchie ist nur durch window.parent abklapperbar aus Sicht der geöffneten Dokumente

Folgendes funktioniert vom dokument, das window.open() hat, im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(){alert;}
var X87=window.open(...);
var X88=X87.document.createElement('DIV');
var X89=X87.document.body.appendChild(X88);
X89.innerHTML='Test'; // DIV erscheint im geöffneten Dokument
X87.document.body.onunload=Y_unload; // alert mit unload des geöffneten Dokumentes
```

Achtung: document.body.onunload funktioniert teilweise nicht mehr !

Blockieren aktiver Inhalte ab IE 6 z.B. als Standardeinstellung.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen
Blockierte Scripte aktivier <NOSCRIPT> falls kodiert (z.B. zum Wechsel auf scriptfreie Seite)

Für ein per window.open() geöffnetes Fenster mit Script gilt:

Hat ein per window.open()geöffnetes Dokument Script im Quelltext, kann das Fenster eventuelle mit Aktivierung des Scriptes (als Aktiver Inhalt) geschlossen werden, wobei dann onunload ausgelöst wird.

Bsp.: Quelltext im Fenster öffnenden Dokument:

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87='parent.Y_unload(0);'; X86[0]=new Function("X87);

....

X85[0].document.body.onunload=X86[0];

// onunload für geöffnetes Dokument festlegen
```

Man prüfe, ob mit Abschaltung der Scriptblockierung im geöffneten Fenster diese geschlossen wird.

Besonderheiten zum IE 6:

immer sichtbar sind
window title bar
status bar 20-25 pixels Höhe
unsichtbares Fenster nicht möglich
window.open wird von Restriktionen beeinflusst:

sFeatures bestimmt interface elements
Wertebereiche sind für
.left >=0
.top >=0
.right >=0



aber autom. Wertekorrektur falls ohne Korrektur nicht sichtbar wären
 window.title bar
 status bar
 window.moveTo
 iX >=0
 iY >=0
 window.moveBy
 iX >=0
 iY >=0
 per Script kein Kios-Mode möglich (F11-Taste im IE)
 window.createPopup
 pop-up erscheint innerhalb der Grenzen des Elternfensters: zIndex nicht änderbar
 iX >=0 relativ zum Elternelement bzw. Desktop
 iY >=0 relativ zum Elternelement bzw. Desktop
 genau 1 Popup zu jedem Zeitpunkt nur aktivierbar (keine parallelen)

Referenz auf Elternfenster, das per window.open() ein Fenster öffnet:

window.opener und window.parent nur bei FRAME oder IFRAME verfügbar
 document.parentWindow immer verfügbar

Wird mit open() ein Dokument geöffnet, das mit replace() im selben Fenster die Webseite wechselt,
 dann wird beim Wechsel der opener gewechselt

Bsp: Dokument mit open()

opENZEIGER.onunload=routine;

wobei routine z.B. das geöffnete Fenster schliesst

Dokument mit replace():

opENZEIGER.onunload wird aktiv mit Wechsel per replace() im selben Fenster

z.B. das geöffnete Fenster schliesst sich

Damit gilt: opENZEIGER.onunload gilt nicht für die Wechselfseite

open() liefert Zeiger auf das window-objekt, sobald das Objekt im DOM ist.

Ein vorhandener Zeiger signalisiert NICHT, dass Kinder von window
 instanziiert bzw. im readyState auf complete sind

Wird per open() ein HTML-Dokument, das mit FRAME enthält, geöffnet, ist
 der Zeiger auf das window-Objekt futsch.

z.B. zeiger_aus_open.onunload=....

Event ist nicht das Event im geöffneten Dokument mit FRAME,
 sondern es wird wegen FRAME überschrieben (mit null wenn
 im Dokument mit FRAME nicht neu definiert)

Es besteht keine Möglichkeit, vom opener im Dokument mit FRAME
 den onunload-Handler zu setzen

z.B. Aufruf einer Routine des Elterndokumentes

```
GlobalerOpenZeiger=window.open(...)
Kette='document.parentWindow.routine_aus_Elternfenster()';
// Funktionsrumpf bilden
// routine_aus_Elternfenster() muss global existieren
// nicht window.opener da nur bei FRAME oder IFRAME verfügbar
// nicht window.parent da nicht verfügbar
GlobalerRoutinenZeiger=new Function("Kette");
// Funktion nur aus Rumpf dynamisch erzeugen
GlobalerOpenZeiger.document.body.onunload=GlobalerRoutinenZeiger;
// und Handler zuweisen
// Achtung: document.body.onunload funktioniert teilweise nicht mehr !
// Achtung: Die onunload-Routine läuft IMMER im Kontext des Elternfensters !!!!
// Zeiger auf das geöffnete Fenster muss also
// bereits in der onunload-Routine enthalten sein,
// wenn Aktionen NUR für das geöffnete Fenster
// vollzogen werden sollen
// Beispiel: window.event als Referenz in der onunload-Routine
// ist immer Event des Elternfensters, das
// im geöffneten Fenster aber nicht verfügbar ist
```




```
//
//
//      zu new Function:
//
//      Zeigerwerte sind nicht als Strings erzeugbar, da es keine
//      Funktion gibt, die aus String den Zeigerwert liefert
//      Das gilt also auch für Zeiger auf z.B. ein Feldalement
//      Abhilfe:
//      Zeigervariable als Bezeichner kodieren und
//      per eval() in Zeiger konvertieren
```

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show()erzeugt.



ein weiteres Fenster (Register) z.B. leere Seite (about:blank)
 beide (Register) liegen in einer gemeinsamen IE-Instanz
 Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,
 bricht der Browser das Dokument mit .show() ab (Scriptfehler).
 Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende
 Meldung angezeigt (in der Informationsleiste):
 'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'
 Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:
 Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren
 oder Pops temporär zulassen, indem Sie auf die Informationsleiste klicken.
 Die Realität zur obigen Meldung ist völlig anders:
 Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter
 Popupblocker einschalten
 weitere Informationen
 jedoch keine Möglichkeit wie laut Bedeutung
 Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.
 Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster
 einer Windowsanwendung z.B. einer anderen IE-Instanz)
 Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.
 Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer
 anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.
 Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen
 Webseiten (die nicht das Popup erzeugt haben).
 Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.
 Der Popupblockerfehler verändert die Eventverwaltung:
 Es werden u.a. ignoriert
 onfocus
 onblur
 onfocusin
 onfocusout
 und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.
 // nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
 window.focus();
 window.document.focus();
 if(document.body!=null)
 {if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
 }
 // wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
 popupzeiger.show(...);

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von
 Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtsstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft,
 wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLLET,
 EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über
 das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.
 Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:



Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tablular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
 (http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)



verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:•

<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab



Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: •



<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>
 (http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm)

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87=parent.Y_unload(0); X86[0]=new Function("X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet,
so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr
wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
  unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild

<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12)">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12)">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,
innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

Attribute

closed Retrieves whether the referenced window is closed.

defaultStatus Sets or retrieves the default message displayed in the status bar at the bottom of the window.

dialogArguments Retrieves the variable or array of variables passed into the modal dialog window.

dialogHeight Sets or retrieves the height of the modal dialog window.

dialogLeft Sets or retrieves the left coordinate of the modal dialog window.

dialogTop Sets or retrieves the top coordinate of the modal dialog window.

dialogWidth Sets or retrieves the width of the modal dialog window.

frameElement Retrieves the frame or iframe object that is hosting the window in the parent document.

length Sets or retrieves the number of objects in a collection.

name Sets or retrieves a value that indicates the window name.

offscreenBuffering Sets or retrieves whether objects are drawn offscreen before being made visible to the user.

opener Sets or retrieves a reference to the window that created the current window.

parent Retrieves the parent of the window in the object hierarchy.

returnValue Sets or retrieves the value returned from the modal dialog window.

screenLeft Retrieves the x-coordinate of the upper left-hand corner of the browser's client area, relative to the upper left-hand corner of the screen.

screenTop Retrieves the y-coordinate of the top corner of the browser's client area, relative to the top corner of the screen.

self Retrieves a reference to the current window or frame.

status Sets or retrieves the message in the status bar at the bottom of the window.

top Retrieves the topmost ancestor window.

Events



onactivate Event erzeugt, wenn Objekt das aktive Objekt wird
 onafterprint Event erzeugt direkt nach Ende des Druckes bzw. Druckvorschau zum Objekt
 onbeforedeactivate Event erzeugt direkt bevor Objekt nicht mehr aktives Objekt wird
 onbeforeprint Event erzeugt direkt vor Beginn des Druckes bzw. Druckvorschau zum Objekt
 onbeforeunload Event erzeugt direkt vor Entladen des Dokumentes
 onblur Event erzeugt, wenn Objekt den Focus verliert (und damit auch nicht mehr aktiv ist)
 oncontrolselect Event erzeugt, wenn User eine Control-Selektion auf das Objekt ausführt
 ondeactivate Event erzeugt, wenn Objekt inaktives Objekt wird
 onerror Fires when an error occurs during object loading.
 onfocus Event erzeugt, wenn Objekt den Focus erhält (und damit auch aktiv wird)
 onhelp Event erzeugt, wenn User die F1-Taste im Browser drückt, um die Hilfe zu aktivieren
 onload Event erzeugt mit Beginn des Ladens des Objektes durch den Browser (Status siehe readyState)
 onmove Event erzeugt, wenn Objekt bewegt wird
 onmoveend Event erzeugt, wenn Bewegung eines Objektes endet
 onmovestart Event erzeugt, wenn Bewegung eines Objektes beginnt
 onresize Event erzeugt, wenn sich die Grösse (Dimension) des Objektes ändert
 onresizing Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes endet
 onresizestart Event erzeugt, wenn die Änderung der Grösse (Dimension) des Objektes beginnt
 onscroll Event erzeugt, wenn User die Scrollbar des Objektes benutzt (Scrollleisten)
 onunload Event erzeugt, wenn Objekt entladen ist (aus Browser entladen)

Methoden

alert Displays a dialog box containing an application-defined message.
 attachEvent Ein Eventhandler für ein Event in das Objekt einbinden und die Eventüberwachung per Handler aktivieren
 blur Dem Objekt den Focus wegnehmen (Objekt wird auch inaktiv)
 clearInterval Cancels the interval previously started using the setInterval method.
 clearTimeout Cancels a time-out that was set with the setTimeout method.
 close Closes the current browser window or HTML Application (HTA).
 confirm Displays a confirmation dialog box that contains an optional message as well as OK and Cancel buttons.
 createPopup Creates a popup window.
 detachEvent Eine für das Objekt per attachEvent() eingebundene Eventüberwachung deaktivieren und Eventhandler aus Objekt entfernen
 execScript Executes the specified script in the provided language.
 focus Dem Objekt den Focus geben (und das Objekt damit aktiv setzen)
 moveBy Moves the screen position of the window by the specified x and y offset values.
 moveTo Moves the screen position of the upper-left corner of the window to the specified x and y position.
 navigate Loads the specified URL to the current window.
 open Opens a new window and loads the document specified by a given URL.
 print Prints the document associated with the window.
 prompt Displays a dialog box that prompts the user with a message and an input field.
 resizeBy Changes the current size of the window by the specified x- and y-offset.
 resizeTo Sets the size of the window to the specified width and height values.
 scroll Causes the window to scroll to the specified x- and y-offset at the upper-left corner of the window.
 scrollBy Causes the window to scroll relative to the current scrolled position by the specified x- and y-pixel offset.
 scrollTo Scrolls the window to the specified x- and y-offset.
 setActive Objekt aktiv setzen, aber nicht Focus geben
 setInterval Evaluates an expression each time a specified number of milliseconds has elapsed.
 setTimeout Evaluates an expression after a specified number of milliseconds has elapsed.
 showHelp Displays a Help file. This method can be used with Microsoft HTML Help.
 showModalDialog Creates a modal dialog box that displays the specified HTML document.
 showModelessDialog Creates a modeless dialog box that displays the specified HTML document.

Styles

keine

Collectionen

frames Retrieves a collection of all window objects defined by the given document or defined by the document associated with the given window.

Objekte von window

clientInformation Contains information about the Web browser.
 clipboardData Provides access to predefined clipboard formats for use in editing operations.
 document Represents the HTML document in a given browser window.
 event Represents the state of an event, such as the element in which the event occurred, the state of the keyboard keys, the location of the mouse, and the state of the mouse buttons.
 external Allows access to an additional object model provided by host applications of the Microsoft® Internet Explorer browser components.
 history Contains information about the URLs visited by the client.
 location Contains information about the current URL.
 navigator Contains information about the Web browser.
 screen Contains information about the client's screen and rendering capabilities.

4.3.2.2.12.9.46. XMLHttpRequest

siehe auch Objekt XML und userProfile

Attribute

readyState Retrieves the current state of the request operation.
 responseBody Retrieves the response body as an array of unsigned bytes.
 responseText Retrieves the response body as a string.
 responseXML Retrieves the response body as an XML Document Object Model (DOM) object.
 status Retrieves the HTTP status code of the request.



statusText Retrieves the friendly HTTP status of the request.

Events

onreadystatechange listener of status of asynchronous request.

Methoden

abort Cancels the current HTTP request.

getAllResponseHeaders Returns the complete list of response headers.

getResponseHeader Returns the specified response header.

open Assigns method, destination URL, and other optional attributes of a pending request.

send Sends an HTTP request to the server and receives a response.

setRequestHeader Adds custom HTTP headers to the request.

Styles

keine

Kurzbeschreibung des Objektes

Das Objekt dient dem Laden und Versenden von XML-Daten über Http-Request vom / zum Server der Webseite:

Asynchroner oder synchroner Zugriff (Request)

XML-Daten können nur per XML-DOM clientseitig in renderbare HTML-Daten konvertiert werden. Alternativ ist Extensible Stylesheet Language Transformations (XSLT) nutzbar.

XML-Daten

sind inzwischen trotz diverser Konventionen ein Austauschformat zwischen Anwendungen (nicht nur unter Windows) geworden.
(daher auch oft kostenpflichtige Software)

sind nicht gerade platzsparend erzeugbar dafür als schlichte Textdatei

sind eventuell Konkurrenz z.B. zu Adobe PDF

sind im Syntax exakter zu verarbeiten als z.B. die (zulässigen) HTML-Tag-Verstümmelungen (in Programmierer-Faulheit)
(allerdings ist das kein Vorteil, sondern das Ausmerzen von unnötigen Unzulässigkeiten und Sicherheitsproblemen).

Eternobjekt ist window.

XMLHttpRequest wird erst ab IE 7 unterstützt (in IE 6 musste Active-X erhalten)

```
var xmlHttp;
if (window.XMLHttpRequest) {xmlHttp = new XMLHttpRequest();} // IE7, Mozilla, Safari, etc.
else
{if (window.ActiveXObject) {xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");}} // IE 5 bis 6
```

Beispiel

```
if (window.XMLHttpRequest)
{
    var ReqObjekt = new XMLHttpRequest();
    ReqObjekt.open("GET", "http://localhost/test.xml");
    ReqObjekt.send();
    alert(ReqObjekt.statusText);
}
```

onreadystatechange

Status des Request überwachen

```
XMLHttpRequest.onreadystatechange=zeiger_auf_handler;
```

lesen und schreiben

Beispiel:

```
function EventHandler_complete ()
{if (ReqObjekt.readyState == 4)alert('Daten geladen.')}

var ReqObjekt = new XMLHttpRequest();
ReqObjekt.onreadystatechange = EventHandler_complete; // ohne '()' !!
ReqObjekt.open("GET", "http://localhost/test.xml", true);
ReqObjekt.send();
```

readyState

Status des Request

```
[ nState = ] XMLHttpRequest.readyState
```

nState	Integer	
0		Objekt ist zwar erzeugt aber nicht initialisiert (Methode open noch nicht aktiviert worden)
1		Methode open ist aktiviert worden, Methode send nicht aktiviert worden
2		Methode send ist aktiviert worden, Status und Headers noch nicht verfügbar
3		Daten wurden empfangen, aber Status und Headers sind nicht vollständig verfügbar
4		Alle Daten sind verfügbar (Objekt komplett initialisiert) (nicht 'complete' verwenden)

nur lesen



responseBody

Daten des Antwort-Body als ein Feld vorzeichenloser Bytes holen

```
[ vBody = ] XMLHttpRequest.responseBody
```

vBody Zeiger auf Datenfeld

nur lesen

responseText

Daten des Antwort-Body als ein String

```
[ sBody = ] XMLHttpRequest.responseText
```

sBody Zeiger auf String

nur lesen

responseXML

Daten des Antwort-Body als ein XML-DOM-Objekt

```
[ oBody = ] XMLHttpRequest.responseXML
```

oBody Zeiger auf Object

nur lesen

Beispiel:

```
var ReqObjekt = new XMLHttpRequest();
ReqObjekt.open("GET", "http://localhost/test.xml",false);
ReqObjekt.send();
alert(ReqObjekt.responseXML.xml);
```

status

numerischer HTTP Status des Request

```
[ nStatus = ] XMLHttpRequest.status
```

nur lesen

nStatus Integer

100	Continue
101	Switching protocols
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
307	Temporary Redirect
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request-URI Too Long
415	Unsupported Media Type



416	Requested Range Not Suitable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported

Beispiel:

```
var ReqObjekt = new XMLHttpRequest();
ReqObjekt.open("GET", "http://localhost/test.xml",false);
ReqObjekt.send();
if (ReqObjekt.status == 401){alert('Access denied.')}
else{alert(ReqObjekt.responseText);}
```

statusText

Zeichenketten-HTTP Status des Request

[sStatus =] XMLHttpRequest.statusText

sStatus siehe .status

nur lesen

Beispiel:

```
var ReqObjekt = new XMLHttpRequest();
ReqObjekt.open("GET", "http://localhost/test.xml",false);
ReqObjekt.send();
if (ReqObjekt.status == 'OK'){alert(ReqObjekt.responseText);}
else{alert(ReqObjekt.statusText);}
```

abort()

aktuellen Request abbrechen

liefert nichts

getAllResponseHeaders()

komplette Liste der Antwort-Header liefern

sHeaders = XMLHttpRequest.getAllResponseHeaders()

String als Liste

Listenaufbau:

Folge von Paaren aus Name und Wert

Paare getrennt durch carriage return/line feed (CR/LF)

getResponseHeader()

Wert eines Antwort-Header liefern

p = XMLHttpRequest.getResponseHeader(bstrHeader)

bstrHeader String, Header-Name

open()

Request öffnen, ermöglicht danach send()

XMLHttpRequest.open(sMethod, sUrl [, bAsync] [, sUser] [, sPassword])

sMethod String, HTTP Methode

'GET'

'POST'

'HEAD'

'PUT'

'DELETE'

'MOVE'

'PROPFIND'

'PROPPATCH'

'MKCOL'

'COPY'



'LOCK'
'UNLOCK'
'OPTIONS'

Gross-kleinschreibung egal

sUrl String, Url (absolut oder relativ) der XML-Daten oder des server-seitigen XML-Web-Services.
alle Urls müssen identische Domain, Port und Protokoll nutzen wie die Webseite, die XML-Daten per XMLHttpRequest-Objekt nutzen will

bAsync optional.
true asynchroner Request
Eventhandler für onreadystatechange verwenden für Statusabfrage.
false nicht asynchroner Request

sUser optional
String Name des Users für Authentifizierung (wenn diese verlangt wird)
wenn Leerkette so Login-Fenster angezeigt

sPassword optional
String Password des Users für Authentifizierung (wenn diese verlangt wird)
nur ausgewertet wenn sUser nicht Leerkette ist (da ansonsten durch Login-Fenster bedient)

liefert nichts

Internet Explorer legt Daten aus GET immer in den Browsercache ab
aus POST niemals in den Browsercache ab

send()

Request senden und Antwort empfangen
benötigt irgendwann open() zuvor

XMLHttpRequest.send([varBody])

varBody optional, Body der Sendedaten z.B. String, Feld vorzeichenloser Bytes oder XML-DOM-Objekt

liefert nichts

setRequestHeader

benutzerdefinierten Header für nächsten Request per send() bereitstellen

XMLHttpRequest.setRequestHeader(sName, sValue)

sName Required. String that specifies the header name.

(Teil 1 des Paares)

sValue Required. String that specifies the header value.

(Teil 2 des Paares)

liefert nichts

Beispiel:

```
var ReqObjekt = new XMLHttpRequest();
....
ReqObjekt.open("POST", sUrl, false);
....
ReqObjekt.setRequestHeader("Content-Type", "text/xml");
....
ReqObjekt.send(sRequestBody);
```

4.3.2.2.12.10. Vordefinierte Farbbezeichner

aliceblue	#F0F8FF
antiquewhite	#FAEBD7
aqua	#00FFFF
aquamarine	#7FFFD4
azure	#F0FFFF
beige	#F5F5DC
bisque	#FFE4C4
black	#000000
blanchedalmond	#FFEBCD
blue	#0000FF
blueviolet	#8A2BE2
brown	#A52A2A
burlywood	#DEB887
cadetblue	#5F9EA0
chartreuse	#7FFF00
chocolate	#D2691E
coral	#FF7F50
cornflowerblue	#6495ED
cornsilk	#FFF8DC



crimson	#DC143C	
cyan	#00FFFF	
darkblue	#00008B	
darkcyan	#008B8B	
darkgoldenrod	#B8860B	
darkgray	#A9A9A9	
darkgreen	#006400	
darkkhaki	#BDB76B	
darkmagenta	#8B008B	
darkolivegreen	#556B2F	
darkorange	#FF8C00	
darkorchid	#9932CC	
darkred	#8B0000	
darksalmon	#E9967A	
darkseagreen	#8FBC8B	
darkslateblue	#483D8B	
darkslategray	#2F4F4F	
darkturquoise	#00CED1	
darkviolet	#9400D3	
deeppink	#FF1493	
deepskyblue	#00BFFF	
dimgray	#696969	
dodgerblue	#1E90FF	
firebrick	#B22222	
floralwhite	#FFFAF0	
forestgreen	#228B22	
fuchsia	#FF00FF	
gainsboro	#DCDCDC	
ghostwhite	#F8F8FF	
gold	#FFD700	
goldenrod	#DAA520	
gray	#808080	
green	#008000	
greenyellow	#ADFF2F	
honeydew	#F0FFF0	
hotpink	#FF69B4	
indianred	#CD5C5C	
indigo	#4B0082	
ivory	#FFFFFF	
khaki	#F0E68C	
lavender	#E6E6FA	
lavenderblush	#FFF0F5	
lawngreen	#7CFC00	
lemonchiffon	#FFFACD	
lightblue	#ADD8E6	
lightcoral	#F08080	
lightcyan	#E0FFFF	
lightgoldenrodyellow	#FAFAD2	
lightgreen	#90EE90	
lightgrey	#D3D3D3	
lightpink	#FFB6C1	
lightsalmon	#FFA07A	
lightseagreen	#20B2AA	
lightskyblue	#87CEFA	
lightslategray	#778899	
lightsteelblue	#B0C4DE	
lightyellow	#FFFFE0	
lime	#00FF00	
limegreen	#32CD32	
linen	#FAF0E6	
magenta	#FF00FF	
maroon	#800000	
mediumaquamarine	#66CDAA	
mediumblue	#0000CD	
mediumorchid	#BA55D3	
mediumpurple	#9370DB	
mediumseagreen	#3CB371	
mediumslateblue	#7B68EE	
mediumspringgreen	#00FA9A	
mediumturquoise	#48D1CC	
mediumvioletred	#C71585	
midnightblue	#191970	
mintcream	#F5FFFA	
mistyrose	#FFE4E1	



moccasin	#FFE4B5
navajowhite	#FFDEAD
navy	#000080
oldlace	#FDF5E6
olive	#808000
olivedrab	#6B8E23
orange	#FFA500
orangered	#FF4500
orchid	#DA70D6
palegoldenrod	#EEE8AA
palegreen	#98FB98
paleturquoise	#AFEEEE
palevioletred	#DB7093
papayawhip	#FFefd5
peachpuff	#FFDAB9
peru	#CD853F
pink	#FFC0CB
plum	#DDA0DD
powderblue	#B0E0E6
purple	#800080
red	#FF0000
rosybrown	#BC8F8F
royalblue	#4169E1
saddlebrown	#8B4513
salmon	#FA8072
sandybrown	#F4A460
seagreen	#2E8B57
seashell	#FFF5EE
sienna	#A0522D
silver	#C0C0C0
skyblue	#87CEEB
slateblue	#6A5ACD
slategray	#708090
snow	#FFFAFA
springgreen	#00FF7F
steelblue	#4682B4
tan	#D2B48C
teal	#008080
thistle	#D8BFD8
tomato	#FF6347
turquoise	#40E0D0
violet	#EE82EE
wheat	#F5DEB3
white	#FFFFFF
whitesmoke	#F5F5F5
yellow	#FFFF00
yellowgreen	#9ACD32

5. Plugins des Netscape und ActiveX-Controls des Internet Explorers für die Integration von Browsererweiterungen

Aufgrund der Desingunterschiede zwischen diesen Browsern muss für die Verwendung eines Plugins wie Shockwave oder Webspeech unterschiedlich programmiert werden.

Nur Netscape unterstützt das Objekt Plugin. Der Internet Explorer beruft sich auf ActiveX als Betriebssystemkomponente. Bezweckt wird allerdings dasselbe: Die Integration fremder Tools in den jeweiligen Browser.

5.1. *Plugins des Netscape für Browsererweiterungen durch Fremdanbieter*

Netscape unterstützt den <EMBED>-Tag (siehe HTML-Beschreibung).

5.2. *ActiveX des Internet Explorer für Browsererweiterungen (z.T. Fremdanbieter)*

Beispiel für prüfen auf ActiveX beim IE:

```
var ActiveXAktiv = false;

var NavigatorObjekt = window.navigator;           // Zeiger

var BrowserArt      = NavigatorObjekt.userAgent;   // String
var IEerkannt       = (BrowserArt.indexOf('IE') > -1); // true, so IE erkannt

var Plattform       = NavigatorObjekt.platform;    // String
var Win32Erkannt    = (Plattform == "Win32");      // true, so Win32-Bit erkannt
```



```
ActiveXAktiv = (IEerkannt && Win32Erkannt); // true, so IE und Win32-Bit erkannt,
// also ActiveX möglich
```

5.2.1. Datenbank im Internet Explorer ab 4.x

ActiveX muss aktiv sein !

Die Textdatenbank wird komplett in den Cache des Users übertragen und ist somit nicht geschützt !

Erst wenn die gesamte Datenbank im Cache geladen ist, wird sie angezeigt !

Die Domain der Datenbank-Datei und des HTML-Dokumentes, das die Datenbank verwendet, müssen identisch sein.

Datenbankverwaltung erfolgt mit dem TDC (Tabular Data Container) als ActiveX-Control

clsid:333C7BC4-460F-11D0-BC04-0080C7055A83

Hinweis: 0 ist Null und nicht Zeichen O

5.2.1.1. Aufbau der Datenbank

Datenbank im Textformat

Datenbank-Datei: Textdatei
satzweise

Datenbank-Datei-Satz: CSV-Format (Comma Separated Values)

endet standardgemäß mit Zeilenumbruch (üblich bei Textdateien \$0D\$0A also newline)

(Satztrenner newline ist abänderbar)

Folge von Feldwerten, die standardgemäß mit Komma zu trennen sind

(Feldtrenner Komma ist abänderbar)

Feldwert-Format: wert[:typ] :typ ist optional

Feldwert-Typen :typ sind zu kodieren im ersten Satz der Textdatei (Daten ab 2. Satz), wenn es sich nicht um String-Typen handelt

Bsp.: Sind alle Daten im String-Typ, so kann 1. Satz bereits Daten enthalten

Empfehlung: Wenn Datenbank sortiert werden soll, so immer

1. Satz mit Feldbezeichnern erzeugen

String ist Standard

kein " " oder ' ' kodieren (Ausnahme: siehe unten Feldtrenner und Blank)

Datexxx mit xxx als Buchstabenfolge von D, M und Y in beliebiger Reihenfolge

Boolean Yes oder No oder True oder False oder 0 oder not 0 wobei Yes identisch zu True

zu Wert > 0

No identisch zu False

zu Wert gleich 0

Int ganze Zahl

Float Gleitzahl

mit Dezimalpunkt laut Windowseinstellung

Deutsch: Punkt für Tausendertrenner

Komma für Dezimalkomma

Englisch (Standard bei JScript):

Komma für Tausendertrenner

Punkt für Dezimalkomma

Feldwert darf nicht enthalten

Feldtrenner und/oder Blank, es sei denn, der Wert des Feldes wird in " "

eingeschlossen und damit Blank bzw. Feldtrenner in ihrer Funktion

entwertet (Entwertungszeichen " ist abänderbar)

deutsche Umlaute und Sonderzeichen

Beispiel für Daten nur vom String-Typ:

ab 1. Satz Guericke,"Otto, von",0815/1234

Mueller,Anton,0815/6789

Beispiel für Daten nur vom String-Typ:

Satz 1 Feldbezeichner

vorname,name,telefon

ab 2. Satz Guericke,"Otto, von",0815/1234

Mueller,Anton,0815/6789

Beispiel für Daten mit Nicht-String-Typ:

1. Satz muss kann Feldbezeichnerliste sein mit Typangabe (außer bei String-Typ)

NahrungsmittelArt,Preis:Float,Kaufdatum:DateYMD,Menge:Int

ab 2. Satz Brot,1.57,97/5/12,30

Kaese,3.52,96/2/2,5

"Alter Wein",183.99,1905-1-1,1



5.2.1.2. HTML-Einbindung**5.2.1.2.1. Objekt-Deklaration**

<BODY>

```

<OBJECT ID="ID_Datenbank" CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83">
    <!-- // 0 ist Null und nicht Zeichen 0 //-->

    <PARAM NAME="DataURL" VALUE="adress.txt">
        <!-- Url und Name der Textdatei, also
                physischer Datenbankname wie z.B.
                file:// oder http:// oder data/elements.txt
                Achtung: Die Domain von Datenbank undHTML-Seite
                müssen identischsein !
                im Beispiel: adress.txt im gleichen Verzeichnis
        //-->

    <PARAM NAME="UseHeader" VALUE="True" oder "False" >
        <!-- True, so 1. Satz nicht anzeigen
                verwenden, wenn 1. Satz mit Feldbezeichner
                False, so 1. Satz anzeigen
                IE prüft nicht, ob 1. Satz mit Feldbezeichner,
                da dem IE die Dateninhalte egal sind.
        //-->

    // nachfolgender PARAM ist optional
    <PARAM NAME="FieldDelim" VALUE="&#09;" >
        <!-- anstelle des Komma als Feldtrenner nun TAB verwenden
                ansonsten einfach das Zeichen kodieren z.B. ";"
        //-->

    // nachfolgender PARAM ist optional
    <PARAM NAME="Filter" VALUE="(logische_verknuepfung_von_feldbezeichnern)">
        <!-- Bsp. "(PhoneNum <> &quot;&quot;)"
                Achtung: ES MUSS geklammert werden.
                logische Verknüpfung wie bei JScript
        //-->

</OBJECT>
</BODY>

```

5.2.1.2.2. Datenfeld-Deklaration

im HTML-Tag, das das Datum verwenden möchte: Nicht alle Tags werden vom TDC unterstützt !
per Anker auf die Datenbank

Beispiel für Tabelle:

Prinzip: jedes Feld wird als HTML-Tabellenspalte dargestellt:
pro Spaltenzeile der Feldinhalt aus dem jeweiligen Satz
Anzahl der Zeilen=Anzahl der Sätze (abgesehen von Feldbezeichnersatz also Satz 1)

Achtung: Nicht alle Tabellenattribute werden von TDC unterstützt !

unterstützt werden: DIV
SPAN
IMG

adress.txt:

1. Satz **vorname;name;telefon**
ab 2. Satz Guericke;"Otto, von";0815
Willmann;Theo;1234
Xantippe;Isa;5678
Arktin;Richard;1055

<BODY>

```

<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE DATASRC=#ID_Datenbank>
<!-- 1. Satz als Kopf //-->
<THEAD>
<TR>

```




```

        <TH>Vorname</TH>
        <TH>Name</TH>
        <TH>Telefon</TH>
    </TR>
</TR>
</THEAD>
<!-- genau 1 Datenzeile kodieren, die als Vorlage für jeden Satz der adress.txt verwendet wird:
      Jeder Satz erhält automatisch in der Tabelle eine eigene Zeile
      adress.txt wird angezeigt in der Satzfolge laut der Datei
-->
<TBODY>
<TR>
    <TD><DIV DATAFLD="vorname"></DIV></TD>
    <TD><DIV DATAFLD="name"></DIV></TD>
    <TD><DIV DATAFLD="telefon"></DIV></TD>
</TR>
</TBODY>
</TABLE>
</BODY>

```

5.2.1.3. Operationen mit der Datenbank

5.2.1.3.1. Datenbank-Objekt

Syntax:

ID_Datenbank.eigenschaft
ID_Datenbank.methode()

Eigenschaften:

.AppendData	Boolean true, so neue Daten anhängen false, so neue Daten ersetzen alte Daten
.CaseSensitive	Boolean true, so Unterscheidung von Gross und Klein false, so keine Unterscheidung von Gross und Klein
.CharSet	Wert für Zeichensatz 20106 DIN 66003 IA5 German 1250 windows-1250 Central European Alphabet 1252 iso-8859-1 und windows-1252, Western Alphabet 50000 x-user-defined
.EscapeChar	Entwertungszeichen nicht in " " bzw. ' ' kodieren Standard ist Leerkette Bsp. für Zeichen innerhalb einer Kette, das identisch mit dem Ketten-Kennzeichnungs.Zeichen ID_Datenbank.EscapeChar=" ermöglicht die Kodierung von "Das ist \"echt\" cool!"
.FieldDelim	Feldtrennerzeichen Standard ist "," Bsp.: "	" für TAB überschreibt <PARAM NAME=FieldDelim ... >
.Filter	für Satzselektion per Filter Wert ist "logische_verknuepfung_von_feldbezeichnern" analog zu <PARAM NAME="Filter" ...> überschreibt <PARAM NAME="Filter" ...> Verwendung von Wildcard * möglich z.B. vorname = O* Bsp: ID_Datenbank.Filter=(('vorname = Otto') & ('name =Test'));
.Language	HTML-Code-Sprache Bsp: "eng-us" (Standard)
.recordset	Zeiger auf Objekt der Satzselektion
.RowDelim	Satztrennzeichen Standard ist "newline"
.Sort	"folge_von_feldbezeichnern_mit_semikolontrennung" Bsp: ID_Datenbank.Sort="- vorname; +name" -vorname für absteigende Sortierung +name für aufsteigende Sortierung Sortierung von Text: lexikographisch, also im Alphabet mit Unterscheidung Gross und Klein



numerischen Werten: bei aufsteigend: von minus zu plus
 bei absteigend: von plus zu minus
 Datumswerten in der Form t/m/jjjj
 wenn t und m identisch sind dann erst nach Jahren
 logischen Werten bei aufsteigend: erst alle False, dann alle True
 bei absteigend; erst alle True, dann alle False

Abbruch der Sortierung sobald ein leeres Feld gefunden wird

Kodierung zur Sortierung:

Falls die Datenbank keine Feldbezeichner im 1. Satz besitzt, so müssen frei erfundenen Feld-
 Bezeichner als Platzhalter kodiert werden.

Hinweis: Es kann nicht folgender Fehler wie in MS-Excel passieren:

Markierung einer Spalte anstelle des gesamten Excel-Blattes

dann Daten sortieren ---> alle Sätze sind **nur in der Spalte**
 sortiert und damit **inhaltlich komplett verändert**

.TextQualifier Zeichen der Stringkodierung für Kodierung von Blank bzw. Feldtrenner als Daten
 nicht in " " bzw. ' ' kodieren
 Standard ist "

Methoden:

.recordset() Referenz auf Wert eines Feldes des aktuellen Satzes liefern

Syntax:

```
[ var ZeigerAufDatenFeldWert = ] ID_Datenbank.recordset(Kette);
```

Kette String

Feldbezeichner laut 1. Datensatz der Datenbank

.Reset(); Datenbank aktualisieren (auch in der Ansicht)
 z.B. nach Selektionseinstellungen
 Filtereinstellungen

5.2.1.3.2. Objekt der Satzselektion (recordset)

Recordset ist das Objekt der Satzselektion per Satzzeiger.

Syntax:

ID_Datenbank.recordset.eigenschaft

ID_Datenbank.recordset.methode()

Eigenschaften:

.AbsolutePosition liefert aktuelle Satzposition
 Integer, ab 1

.BOF Boolean
 true wenn Satzzeiger vor dem ersten Satz steht
 false wenn Satzzeiger nicht vor dem ersten Satz steht
 Anwendung in Schleifen

.EOF Boolean
 true, wenn Zeiger hinter dem letzten Satz steht
 false, wenn Zeiger nicht hinter dem letzten Satz steht
 Anwendung in Schleifen

RecordCount Gesamtanzahl der Sätze in der Datenbank
 beachte <PARAM NAME="UseHeader" ...>

Methoden:

.MoveFirst(); ersten Satz der Datenbank einstellen (Satzzeiger auf ersten Satz setzen)
 .MoveNext(); nächsten Satz einstellen und damit Satzzeiger verändern
 .MovePrevious(); vorhergehenden Satz einstellen und damit Satzzeiger verändern

5.2.1.4. Beispiele

5.2.1.4.1. Sortierung

```
adress.txt:
1. Satz  vorname;name;telefon
ab 2. Satz Guericke;"Otto, von";0815
Willmann;Theo;1234
Xantippe;Isa;5678
Arktin;Richard;1055

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
var SortierRichtung = true;
```

```
function Sortieren(FeldBezeichner)
```



```

    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) { Kette = "+"; }

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }
// -->
</SCRIPT>
</HEAD>

<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE DATASRC=#ID_Datenbank>
<THEAD>
<TR>
    <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
    <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
    <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
</TR>
</TR>
</THEAD>
<TBODY>
<TR>
    <TD><DIV DATAFLD="vorname"></DIV></TD>
    <TD><DIV DATAFLD="name"></DIV></TD>
    <TD><DIV DATAFLD="telefon"></DIV></TD>
</TR>
</TBODY>
</TABLE>
</BODY>
</HTML>

```

5.2.1.4.2. Blättern in Datenbank

```

adress.txt:
1. Satz   vorname;name;telefon
ab 2. Satz Guericke;"Otto, von";0815
          Willmann;Theo;1234
          Xantippe;Isa;5678
          Arktin;Richard;1055

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) { Kette = "+"; }

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

```



```

    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition != ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == 1)
        {alert("Erster Datensatz erreicht!");}
    }

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
>
<THEAD>
<TR>
    <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
    <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
    <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
</TR>
</THEAD>
<TBODY>
<TR>
    <TD><DIV DATAFLD="vorname"></DIV></TD>
    <TD><DIV DATAFLD="name"></DIV></TD>
    <TD><DIV DATAFLD="telefon"></DIV></TD>
</TR>
</TBODY>
</TABLE>
<BR>
<INPUT
    TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT
    TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

```



```
</BODY>
</HTML>
```

5.2.1.4.3. Satzselektion mit Filter

```
1.Satz   vorname,name,telefon
2. Satz  Guericke,"Otto, von",0815/1234

function filtern()
{
    ID_Datenbank.Filter='vorname=Otto';
    ID_Datenbank.Reset();
    return(false);
}

<INPUT  TYPE="button"
        VALUE="Filtern nach Vornamen Otto "
        onclick="filtern();"
>
```

5.2.2. Direct Animation im Internet Explorer (Übersicht DA als DirectX-Komponente)

Direct Animation ist das Pendant z.B. zu Flash von Adobe (ursprünglich von Macromedia).

Direct Animation ist im Gegensatz zu Adobe- (Macromedia-) Produkten äußerst preiswert: Null Euro. Direct Animation ist Teil von Windows und dessen Konzeption.

Direct Animation arbeitet ohne Plugins und lässt sich ohne Fremdsoftware programmieren: Nachfolgend eine Übersicht zu Windows DirectAnimation (DA) ab IE 3.x auf Basis von ActiveX-Controls von DirectX am Beispiel der CLASS-ID B6FFC24C-7E13-11D0-9B47-00C04FC2F51D (DirectX-SDK von 1998

Warnung: CLASS-ID **B6FFC24C-7E13-11D0-9B47-00C04FC2F51D** kann bereits unter Windows XP SP2 abgeschaltet sein:

Das Objekt zur Class-ID lässt sich erzeugen, aber die Bibliothek MeterLibrary ist null-Zeiger sein.
Es ist zwingend auf Vorhandensein der Bibliothek zu prüfen !
Des Weiteren muss damit gerechnet werden, dass Microsoft auch andere CLASS-ID abschaltet (siehe unten), so dass einst funktionierende Skripte irgendwann nicht mehr funktionieren.

Unter Windows XP SP1 dürfte die Bibliothek MeterLibrary existieren, da Windows XP SP1 von Microsoft per Definition nicht mehr weitergepflegt wird (außer aus Sicht Microsoft kritischer Patches, wobei Microsoft bereits permanent bescheinigt, dass Win XP SP1 ohne SP2 nicht mehr sicher ist.)

DirectX wird permanent weiterentwickelt und zwar z.Z. nicht komplett abwärtskompatibel:

Ganze DirectX-Bibliotheken werden ersetzt, die dann älteres DirectX nicht mehr unterstützen - warum auch immer. Da DirectX auf Hardwaretreiber, die von Microsoft abgenommen sind, basiert, wird mit jedem neuen Windows, das die Treiberschnittstellen ändert (Win XP SP2 und Win Vista) das Risiko erneuert, dass Webseiten, die DirectX benutzen, im DirectX-Teil nicht mehr laufen werden.
Man beachte Abschaltungen von Windows-Komponenten im Rahmen der Sicherheitspatches von Microsoft (siehe auch unten).

Die Scriptmaschine des IE kann JScript-Code auf Basis von ActiveX-Controls verarbeiten, der Anweisungen von DirectAnimation (DA) enthält.

Die Kodierung des ActiveX-Controls erfolgt per üblichen OBJECT-Tag im BODY-Teil des HTML-Dokumentes.

Für das STYLE-Attribut des OBJECT-Tags ist **dringend anzuraten**, es **nicht** zu kodieren, sondern **alle** Style-Angaben ausschliesslich durch Referenz per `zeiger_auf_da_objekt.style.style_wert =;` zu erzeugen. Grund: Je nach DA-Objekt-Implementierung wird das STYLE-Attribut im OBJECT-Tag teilweise oder vollständig ignoriert und damit wirkungslos. Der Zeiger auf das DA-Objekt ist der Wert des ID-Attributes im OBJECT-Tag oder der Zeiger laut `createElement()` vom OBJECT.

Eine Integration von JScript mit DA ist nur z.T. möglich, da DA ein eigenes Laufzeitsystem besitzt, das grundsätzlich Objekte **von DA** erwartet .

Alle DA-Aktionen werden erst mit dem Start des Laufzeites der DA-Animation abgearbeitet. Dabei ist zu beachten, dass das Laufzeitsystem und seine DA-Aktionen **sehr intensiv** die Timer von Windows nutzen. Es ist daher empfehlenswert, möglichst keine parallelen Timer per Direct Animation und/oder z.B. per Javascript-Anweisung `setTimeout()` zu nutzen. Mit Ablauf der DA-Animation ist dringend das Stoppen des Laufzeites und damit der Timernutzung zu empfehlen ! Beispielsweise kann eine permanente Direct Animation des Hintergrundes das Timing des Browsers und von Windows (z.B. PC-Uhr) stark verzögern bzw. sogar lahmlegen, so dass ein Neustart des Rechnersystems nötig werden kann. DA-Aktionen sind also **sparsam** zu verwenden ! Direct Animation stammt - trotz der noch immer ansehnlichen Animationsmöglichkeiten - aus älteren Zeiten von Windows und dem Internet Explorer. Es ist weiterhin empfehlenswert, schnelle PC-Hardware und ein modernes Windows (z.B. Windows XP) zu verwenden, da dann Timerprobleme weniger auftreten. Allerdings hat nicht jeder User diese Möglichkeiten. Aber auch dann können Timerprobleme auftreten.

Nachteilig ist bezüglich der Direct Animation mit Soundwiedergabe der Umstand, dass in der eventuell windoweigenen Lautstärkeregelung die vom User als dauerhaft eingestellten Reglerstellungen zum Ärgernis des Users temporär oder sogar permanent verändert werden. Eine Synchronisierung mit Lautstärken von per JScript erzeugten Soundobjekten (z.B. `bgSound` Objekt) ist z.T. **nicht** möglich, so dass der User erneut Einstellungen der Regler (je nach Sound-Medium wie Wave oder Midi) als Korrektur treffen muss.

Sämtliche DA-Objekte sind Instanzen einer DA-Bibliothek, die alle je nach Zweck der DA in den JScript-Code eingebunden werden müssen (auch die DA-Bibliothek selbst).



DA-Objekte sind z.B. auch numerische Operationen oder Schleifen.

Beispiel: Addition in JScript per Operator +
Addition in DA per DA-Methode .Add()

Die Programmierung von DA ist sehr gewöhnungsbedürftig, abstrakt und aufwendig.

Es ist zu empfehlen:

VAR-Kodierung von Variablen in JScript einzuhalten (ohne VAR kodierte Variablen sind grundsätzlich global)
Variablenverknüpfungen als Folge von Punktnotationen zu vermeiden und dafür Einzelanweisungen zu kodieren, um die Performance des Browsers zu verbessern.

Beispiel:

```
var DA_Farbe_Red = DA_Bibliothek.Red;  
var DA_FuellFarbe_Red = DA_Bibliothek.SolidColorImage(DA_Farbe_Red);  
  
auch kodierbar: var DA_FuellFarbe_Red = DA_Bibliothek.SolidColorImage(DA_Bibliothek.Red);
```

Nachfolgend wird DA nur in Verbindung mit praktischen Beispielen als Einzellösungen skizziert.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
Klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler



Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()
 Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.
 Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.
 Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

- Scriptfehleranzeige ist erlaubt im IE 7
- Popupblocker ist im IE abgeschaltet
- ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.
- ein weiteres Fenster (Register) z.B. leere Seite (about:blank)
- beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,
 bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende
 Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren
 oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten
 weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster
 einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer
 anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen
 Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus
 onblur
 onfocusin
 onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

```

window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
{document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);

```

Hinweis: Der Populfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von
 Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft,
 wobei für den JScript-Programmierer massive Änderungen eintreten.



Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
 Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•
http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}
 Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop



zuveröffentlichen. Weitere Informationen: • 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen: • 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen: • http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp (http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen: • <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp> (<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement



abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird



daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

5.2.2.1. **DA-Bibliothek**

Beispiel: Start des DA-Control mit Laden des Dokumentes **ohne** Ereignissteuerung

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    function DirectAnimationStart()
    {
        // +++++ DA-Bibliothek als Objekt passend zum ID laut OBJECT-Tag
        var DA_Bibliothek = DAControl.PixelLibrary;

        // hier den DA- und JScript-Kode ablegen
        // im DA-Kode alle Instanzen per Punktnotation von DA_Bibliothek ableiten

        // .....
        DAControl.Start(); // späteres DAControl.Stop() möglich
    }
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
    >
    </OBJECT>
</BODY>
</HTML>
```

Man beachte die Eigenschaften des Objektes Object (nicht Script-Objekt Object).

Beispiel: Start des DA-Control mit Laden des Dokumentes **mit** Ereignissteuerung

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var DirectAnimationStart_Aktiv=false; // muss global sein

    function DirectAnimationStart_OnReadyStateChange_Handler()
    {
        // prüfen ob DA-Animation noch nicht gestartet wurde
        if (!X_Animation_DirectAnimationStart_Aktiv)
        {
            // noch nicht gestartet

            // als gestartet markieren
            DirectAnimationStart_Aktiv=true;

            // und genau 1x starten
            DirectAnimationStart();

            // hier sind weitere globale Variablen belegbar, die zur Steuerung anderer Routinen
            // dienen, die nach Start des DA-Controls aktiv werden sollen, wobei
            // die globalen Variablen in den entsprechenden Routinen für deren Start
            // rekursiv z.B. per setTimeout() auf Wertänderung abgefragt werden müssen
            // (Routinen rufen sich selbst solange rekursiv auf und prüfen nach Zeitintervall, ob
            // sich der Wert der zugehörigen globalen Variable geändert hat, bis
            // die Variable anzeigt, dass die eigentliche Aktion der Routine starten kann
            // und die Rekursion damit nicht mehr erfolgen muss).
```



```

    }

    // wenn gestartet, dann tue nichts, falls sich wieder der Status onready ändert sollte
    // und der Programmierer diese Statusänderungen nicht auswerten will
}

function DirectAnimationStart()
{
    // +++++ DA-Bibliothek als Objekt passend zum ID laut OBJECT-Tag
    var DA_Bibliothek = DAControl.PixelLibrary;

    // hier den DA- und JScript-Kode ablegen
    // im DA-Kode alle Instanzen per Punktnotation von DA_Bibliothek ableiten

    // .....
    DAControl.Start(); // späteres DAControl.Stop(); möglich
}

//-->
</SCRIPT>
</HEAD>
<BODY>
    <OBJECT ID="DAControl"
            CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
            onreadystatechange='DirectAnimationStart _OnReadyStateChange_Handler';
    >
</OBJECT>
</BODY>
</HTML>

```

Start des DA-Controls durch Kodierung aller Funktionen im HEAD des Dokumentes bei leerem BODY-Teil:

Es ist zu beachten, dass der Start des DA-Controls **nur während des Ladens** des Dokumentes erfolgen kann und das Ende-Tag </BODY> noch nicht geparkt sein darf. Damit ist ein Start zu einem beliebigen späteren Zeitpunkt z.B. per onclick-Handler **leider** nicht möglich.

Abhilfe schafft folgende Vorgehensweise:

1. DA-Control instanzieren per OBJECT-Tag (im obigen Beispiel mit ID="**DAControl**")
mit STYLE-Attribut STYLE="visibility:'hidden';"
bzw. hinter der OBJECT-Tag-Deklaration nachträglich mit
DAControl.style.visibility='hidden';
und damit als nicht sichtbar anzeigen lassen
2. Start des DA-Controls während des Ladens des Dokumentes:
DA animiert unsichtbar und nutzt Timer von Windows.
3. Zum Zeitpunkt der erwünschten Sichtbarkeit
DAControl.style.visibility='visible';
abarbeiten lassen.
DA animiert sichtbar und nutzt Timer von Windows.
4. Zum Zeitpunkt der erwünschten Un-Sichtbarkeit
DAControl.style.visibility='hidden';
abarbeiten lassen.
DA animiert unsichtbar und nutzt Timer von Windows.

Die Unsichtbarkeit immer vor dem Stop der DA einstellen, also vor Aktivierung von **DAControl.Stop()**;

Die z.B. Verschiebung des gesamten DA-Controls auf dem Bildschirm lässt sich durch Erzeugung eines DIV realisieren, dessen Kind das DA-Control ist. Es muss nur noch der DIV als Container animiert werden. Die Wahl des DIV's ist dann sinnvoll, wenn das DIV Objekt Eigenschaften zur Animation besitzt, die das Object Objekt (nicht Script-Objekt Object) nicht unterstützt (inklusive Styles, Filter und Ereignisse). Man beachte, dass Eigenschaften z.B. style.visibility vom DIV an das Kind vererbt werden können, also dass das Kind diese Eigenschaften nicht kodiert haben muss, wenn der DIV sie bereits belegt.

5.2.2.2. **DA-Objekte vordefiniert (Auswahl)**

Es folgt eine Auswahl von DA-Objekten, die in DA vordefiniert sind, aber alle per Ableitung aus der DA-Bibliothek instanziiert werden müssen.

5.2.2.2.1 DA-Farben

Farben können mit DA-Objekten verbunden werden.

Es gibt vordefinierte Farben und per DA-Methoden erzeugbare Farben

5.2.2.2.1.1. **DA-Farbe vordefiniert**

Beispiele:

```
var DA_Farbe_Red = DA_Bibliothek.Red;
```



```

var DA_Farbe_Green      = DA_Bibliothek.Green;
var DA_Farbe_Blue      = DA_Bibliothek.Blue;
var DA_Farbe_Purple    = DA_Bibliothek.Purple;
var DA_Farbe_Teal      = DA_Bibliothek.Teal;
var DA_Farbe_Navy      = DA_Bibliothek.Navy;
var DA_Farbe_Maroon    = DA_Bibliothek.Maroon;
var DA_Farbe_Yellow    = DA_Bibliothek.Yellow;

```

5.2.2.2.1.2. DA-Füllfarbe

Füllfarben dienen z.B. zur Füllung der Fläche eines DA-Objektes.

Nur unter dem IE 3.x muss die gesamte DA mit einer Füllfarbe versehen werden.

Beispiel:

```

var DA_Farbe_Red      = DA_Bibliothek.Red;
var DA_FuellFarbe_Red = DA_Bibliothek.SolidColorImage(DA_Farbe_Red);

```

5.2.2.2.2. DA-Linie

Beispiel:

```

var DA_StandardLinie = DA_Bibliothek.DefaultLineStyle;

```

5.2.2.2.3. DA-Event

Beispiel:

```

// ----- linker Maustastendruck als DA-Objekt
var DA_LeftButtonDown = DA_Bibliothek.LeftButtonDown;

```

5.2.2.2.4. DA-Timer

Beispiel:

```

// ----- DA-Endlos-Timer
var DA_EndlosTimer = DA_Bibliothek.LocalTime;

```

5.2.2.2.5. DA-Zahl mit numerischem Wert

Beispiel:

```

function ScriptWert_Nach_DA_Wert_Konvertieren(ScriptWert)
{return DA_Bibliothek.DANumber(ScriptWert);}

```

5.2.2.2.6. DA-Operator

Beispiele:

```

function DA_Multiplikation(DA_Faktor1, DA_Faktor2)
{return DA_Bibliothek.Mul(DA_Faktor1, DA_Faktor2);}

function DA_Division(DA_Dividend, DA_Divisor)
{return DA_Bibliothek.Mul(DA_Dividend, DA_Divisor);}

```

5.2.2.3. Kombination von DA-Objekten anhand von Beispielen

Sämtliche Kombinationen müssen vor dem Start der DA-Animation komplett beschrieben werden. Das gilt auch für Aktionen mit den DA-Objekten. DA-Objekte, Kombinationen und Aktionen sind alle Instanzen der DA-Bibliothek.

5.2.2.3.1. 2D-Objekte in der Ebene bzw. im Raum

Das DA-Koordinatensystem entspricht dem der Vektorrechnung. DA lässt also Vektorrechnung zu und somit eine beliebige Positionierung im Raum. Der Ursprung des DA-Koordinatensystemes liegt in der **Mitte** der Dimension laut OBJECT-Tag.

Das DA-Koordinatensystem hat bezüglich der Y-Achse folgende Regelung:

Y-Achse positive Werte **unterhalb** vom Ursprung
 negative Werte **oberhalb** vom Ursprung

5.2.2.3.1.1. 2D-Objekte ohne Rotation: Geometrische Objekte und Text in der Ebene

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
    // ##### allgemeine Angaben #####
    // +++++ Angaben zur DA-Animation im DA-Koordinatensystem, das über der Ebene des Bereiches laut
    //          OBJECT-Tag liegt (Dimension des Bereiches laut Angaben im OBJECT-Tag)
    var AbstandVomDA_Ursprung_Vertikal    = -240; // Summe der Höhen der 2D-Objekt und der Objektabstände
                                                // bilden, dann Summe dividiert durch 2 und das Ergebnis
                                                // negativ verwenden
                                                // Höhenangaben wurden direkt zum betreffenden 2D-Objekt
                                                //          kodiert

    var AbstandVomDA_Ursprung_Horizontal = 0;

    // +++++ Angaben zur relativen Positionierung der DA-Objekte innerhalb der DA-Animation
    var AbstandDer2DObjekte              = 60;

    // +++++ Angaben zum 2D-Objekt Text +++++

```




```

var FontArt           = "Arial";
var FontHoehe         = 22;

// +++++ Angabe zum Stil der Füllung der 2D-Objekte mit Farbe: Art und Weise der Farbdarstellung
//                                     als Kombination der primären und sekundären Füllfarben
var FillStyle         = 11;      // >=0

// ##### DA-Bibliothek #####
var DA_Bibliothek = DAControl.PixelLibrary;

// ##### diverse DA-Groessen #####
var DA_Farbe_White    = DA_Bibliothek.White;
var DA_Farbe_Red      = DA_Bibliothek.Red;
var DA_Farbe_Blue     = DA_Bibliothek.Blue;

// ##### DA-Animation #####
// +++++ erzeugen +++++
var DA_Animation = DA_Bibliothek.NewDrawingSurface();

// +++++ und im Layout verändern +++++
// ----- Skalierung beim Füllen auf auto
DA_Animation.AutoSizeFillScale();

// ----- Rahmenfarbe
DA_Animation.BorderColor(DA_Farbe_White);

// ----- Linienfarbe
DA_Animation.LineColor(DA_Farbe_White);

// ----- Füllfarbe
// - - - primäre
DA_Animation.FillColor(DA_Farbe_Red);

// - - - sekundäre
DA_Animation.SecondaryFillColor(DA_Farbe_Blue);

// ----- Font für Textdarstellung
DA_Animation.Font(FontArt, FontHoehe, true, false, false, false);

// - - - Style beim Füllen
DA_Animation.FillStyle(FillStyle);

// ##### 2D-Objekte der DA-Animation #####
// +++++ 2D-Objekt Vieleck +++++

// ----- Mittelpunkt des Vieleckes in der DA-Animation festlegen
var DA_Mittelpunkt = DA_Bibliothek.Translate2(    AbstandVomDA_Usprung_Horizontal,
                                                  AbstandVomDA_Usprung_Vertikal
                                                  );

// ----- Status retten wegen Veränderung durch nachfolgendes .Transform()
DA_Animation.SaveGraphicsState();

// ----- Mittelpunkt in die Animation einfügen
DA_Animation.Transform(DA_Mittelpunkt);

// ----- Linien ziehen relativ zum Mittelpunkt und nicht DA-Ursprung
//           X-Achse   positive Werte rechts vom Mittelpunkt
//           negative Werte links vom Mittelpunkt
//           Y-Achse   positive Werte unterhalb vom Mittelpunkt
//           negative Werte oberhalb vom Mittelpunkt
// - - - Koordinaten des Vieleck festlegen
var AchtEckKoordinatenFeld = new Array(-15,5, -5,15, 5,15, 15,5, 15,-5, 5,-15, -5,-15, -15,-5);
// 8 Ecken
// pro Ecke X,Y

// - - - und Vieleck zeichnen
DA_Animation.Polygon(AchtEckKoordinatenFeld);

// ----- geretteten Status laden
DA_Animation.RestoreGraphicsState();

// +++++ 2D-Objekt Rechteck +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal += AbstandDer2DObjekte;

```




```

// ----- und Rechteck zeichnen
DA_Animation.Rect( AbstandVomDA_Usprung_Horizontal,
                   AbstandVomDA_Usprung_Vertikal,
                   50,      // Breite
                   20       // Höhe
                   );

// +++++ 2D-Objekt Rechteck mit runden Ecken +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal +=AbstandDer2DObjekte;

// ----- und Rechteck mit runden Ecken zeichnen
DA_Animation.RoundRect( AbstandVomDA_Usprung_Horizontal,
                        AbstandVomDA_Usprung_Vertikal,
                        50,      // Breite
                        20,      // Höhe
                        10,      // Abzug von der Breite für Bildung der runden Ecken
                                // es wird rechts und links abgezogen
                        10       // Abzug von der Höhe für Bildung der runden Ecken
                                // es wird oben und unten abgezogen
                        );

// +++++ 2D-Objekt Text +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal +=AbstandDer2DObjekte;

// ----- Text zeichnen
DA_Animation.Text( "freier Text",
                  AbstandVomDA_Usprung_Horizontal,
                  AbstandVomDA_Usprung_Vertikal
                  );

// +++++ 2D-Objekt Oval +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal +=AbstandDer2DObjekte;

// ----- Oval zeichnen
DA_Animation.Oval( AbstandVomDA_Usprung_Horizontal,
                  AbstandVomDA_Usprung_Vertikal,
                  50,      // Breite
                  50       // Höhe
                  );

// +++++ 2D-Objekt Kreisbogenfläche +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal +=AbstandDer2DObjekte;

// ----- Kreisbogenfläche zeichnen
DA_Animation.PieDegrees( AbstandVomDA_Usprung_Horizontal,
                        AbstandVomDA_Usprung_Vertikal,
                        -10,      // von Position in Grad, 0 entspricht Osten (3 Uhr)
                                // > 0 so entgegen Uhrzeigersinn
                                // < 0 so im Uhrzeigersinn
                        -120,     // zu Position in Grad, 0 entspricht Osten (3 Uhr)
                                // > 0 so entgegen Uhrzeigersinn
                                // < 0 so im Uhrzeigersinn
                        50,      // Breite
                        50       // Höhe
                        );

// +++++ 2D-Objekt Kreisbogen-Strich +++++
// ----- neue vertikale Position einstellen
AbstandVomDA_Usprung_Vertikal +=AbstandDer2DObjekte;

// ----- Kreisbogen-Strich zeichnen
DA_Animation.ArcDegrees( AbstandVomDA_Usprung_Horizontal,
                        AbstandVomDA_Usprung_Vertikal,
                        -10,      // von Position in Grad, 0 entspricht Osten (3 Uhr)
                                // > 0 so entgegen Uhrzeigersinn
                                // < 0 so im Uhrzeigersinn
                        -120,     // zu Position in Grad, 0 entspricht Osten (3 Uhr)
                                // > 0 so entgegen Uhrzeigersinn
                                // < 0 so im Uhrzeigersinn

```



```

50,      // Breite
50      // Höhe
);

// ##### DA-Animation als Bild erzeugen #####
DA_Animation = DA_Animation.Image;

// ##### DA-Init #####
DAControl.Image = DA_Animation;

// ##### Start der Animation #####
DAControl.Start();
}
-->
</SCRIPT>
</HEAD>
<BODY BGCOLOR="black" onload="DirectAnimationStart();">
    <OBJECT ID="DAControl"
        CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
        STYLE="position:relative;width:800;height:600;"
    >
        </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.1.2. 2D-Objekte mit 2D- und 3D-Rotation auf Basis einer periodischen Sinus-Schwingung

Eine Sinus-Schwingung wird wie folgt beschrieben (Physik):

Die Schwingung erfolgt auf einer Kreisbahn, die im zeitlichen Intervall gestreckt wird. Das Strecken erzeugt eine Berg- und Tal-Ansicht, die auch als Phasenmodell bezeichnet wird.

Ohne die zeitliche Streckung gilt folgender Ansatz:

Kreis:	Kreisumfang	ist $2 * \pi * \text{KreisRadius}$
Sinus:	Phasenursprung	ist $0 * \pi$
	Phasenende	ist $2 * \pi$

Sinus-Schwingung findet auf dem Kreisumfang statt, also ist

eine volle Sinus-Schwingung	= $2 * \pi$
Gesamtanzahl der Sinus-Schwingungen	= $2 * \pi * \text{AnzahlSchwingungen}$
mit $\text{AnzahlSchwingungen} > 0$	

Die zeitliche Streckung erfolgt in DA durch einen periodischen Timer, der das Phasenmodell sozusagen **animiert**.

Als Elongation wird der Punkt auf der Phase bezeichnet, also die Position auf einem Berg oder Tal, also der Abstand bezüglich der Ursprunges auf der Y-Achse im Rahmen des zeitlichen Intervalls.

Ohne zeitliches Intervall ist die Elongation die Position auf dem Kreisumfang, also $\text{Sinus}(2 * \pi * \text{GesamtanzahlSchwingungen})$.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
// ##### allgemeine Angaben #####
// +++++ Dimensionen der 2D-Objekte der Direct Animation +++++
var HoeheBzwBreiteInPixel = 100;      // > 0

// +++++ Sinus-Schwingung der Animation definieren
var AnzahlSchwingungen = .3; // > 0

// +++++ 3D-Rotationsgeschwindigkeit +++++
var Geschwindigkeit_3DRotation = 100; // > 0

// +++++ 3D-Rotationsachsen +++++
var Achse_X_3DRotation = true;          // false für keine Rotation um X-Achse
var Achse_Y_3DRotation = true;          // false für keine Rotation um Y-Achse
var Achse_Z_3DRotation = true;          // false für keine Rotation um Z-Achse

function DirectAnimationStart()
{
    // ##### diverse Scriptgrößen #####
    var Dimension1 = 3 * HoeheBzwBreiteInPixel / 4;
    var Dimension2 = HoeheBzwBreiteInPixel / 2;

    // ##### diverse DAFunktionen #####
    function ScriptWert_Nach_DA_Wert_Konvertieren(ScriptWert)
    {return DA_Bibliothek.DANumber(ScriptWert);}

    function DA_Multiplikation(DA_Faktor1, DA_Faktor2)

```



```

{return DA_Bibliothek.Mul(DA_Faktor1, DA_Faktor2);}

function DA_Division(DA_Dividend, DA_Divisor)
{return DA_Bibliothek.Mul(DA_Dividend, DA_Divisor);}

// ##### DA-Routine zur Ermittlung der Elongation mit zeitlicher Steckung und periodischem Wert
function DA_ElongationErmittleIn()      // per DA-Operationen
{
    // +++++ GesamtAnzahlSchwingungen berechnen und als DA-Objekt erzeugen
    var GesamtAnzahlSchwingungen        = 2 * Math.PI * AnzahlSchwingungen;
    var DA_GesamtAnzahlSchwingungen     =
        ScriptWert_Nach_DA_Wert_Konvertieren(GesamtAnzahlSchwingungen);

    //
    //          Frequenz = Anzahl der Schwingungen in einer Zeiteinheit z.B. Sekunden
    //                      Bsp. 3 Schwingungen pro Sekunde
    //
    //          Kreisfrequenz ist 2 * Pi * Frequenz
    //          also Frequenz ist GesamtAnzahlSchwingungen / Zeiteinheit
    //                      2 * Pi * AnzahlSchwingungen / Zeiteinheit

    // +++++ Elongations-Faktor ermitteln anhand von DA-Operationen
    //
    //          Elongation      aktuelle Position über alle vollen Sinus-Schwingungen
    //                      ist sin( Kreisfrequenz * Zeiteinheit )
    //
    //          wobei          Kreisfrequenz * Zeiteinheit
    //
    //          ist           2 * Pi * Frequenz * Zeiteinheit
    //
    //          ist           2 * Pi * (GesamtanzahlSchwingungen / Zeiteinheit)
    //                      * Zeiteinheit
    //
    //          ist           2 * Pi * GesamtanzahlSchwingungen

    // ----- Schwingungen mit Timer versehen, also periodische Wertänderung
    //          anstelle von 2* Pi wird die zeitliche Streckung per Timer verwendet
    var DA_ElongationsFaktor = DA_Multiplikation(DA_EndlosTimer, DA_GesamtAnzahlSchwingungen);
    // +++++ Elongation liefern
    return DA_Bibliothek.Sin(DA_ElongationsFaktor);
}

// ##### DA-Bibliothek als Objekt #####
var DA_Bibliothek = DAControl.PixelLibrary;

// ##### diverse DA-Größen #####
// +++++ Farben #####
var DA_Farbe_Red      = DA_Bibliothek.Red;
var DA_FuellFarbe_Red = DA_Bibliothek.SolidColorImage(DA_Farbe_Red);

var DA_Farbe_Green    = DA_Bibliothek.Green;
var DA_FuellFarbe_Green = DA_Bibliothek.SolidColorImage(DA_Farbe_Green);

var DA_Farbe_Blue     = DA_Bibliothek.Blue;
var DA_FuellFarbe_Blue = DA_Bibliothek.SolidColorImage(DA_Farbe_Blue);

var DA_Farbe_Purple    = DA_Bibliothek.Purple;
var DA_FuellFarbe_Purple = DA_Bibliothek.SolidColorImage(DA_Farbe_Purple);

// +++++ Standardlinie #####
var DA_StandardLinie = DA_Bibliothek.DefaultLineStyle;

// +++++ Mausevent-Arten #####
// ----- linker Maustastendruck als DA-Objekt
var DA_LeftButtonDown = DA_Bibliothek.LeftButtonDown;

// +++++ DA-Endlos-Timer #####
var DA_EndlosTimer = DA_Bibliothek.LocalTime;

// +++++ DA-Elongation #####
var DA_Elongation = DA_ElongationErmittleIn();

// ##### 2D-Objekte erzeugen #####
// +++++ 2D-Objekt Oval #####
//          Animation erfolgt als 2D-Rotation um den Mittelpunkt einer Kreisbahn
// ----- Oval erzeugen
var DA_Oval = DA_Bibliothek.Oval(HoeheBzwBreiteInPixel, HoeheBzwBreiteInPixel);

// ----- und mit Farbe füllen
DA_Oval = DA_Oval.Fill(DA_StandardLinie, DA_FuellFarbe_Red);

```



```

// ----- Mittelpunkt der Animation des Ovals ermitteln
var DA_Mittelpunkt = DA_Bibliothek.Translate2(Dimension1, Dimension1);

// ----- Radius der Animation des Ovals berechnen bezüglich periodischer Elongation
//          Oval wird entlang des Radius animiert
var DA_Radius = DA_Bibliothek.Scale2UniformAnim(DA_Elongation);

// ----- 2D-Komposition aus Mittelpunkt und Radius erzeugen (Animation des Ovals)
var DA_2DKomposition = DA_Bibliothek.Compose2(DA_Mittelpunkt, DA_Radius);

// ----- Oval in der 2D-Animation positionieren
DA_Oval = DA_Oval.Transform(DA_2DKomposition);

// +++++ 2D-Objekt Rectangle 1 +++++
//          Animation erfolgt anhand einer Koordinatensystem-Achse als Rotationsmittelpunkt
// ----- Rectangle 1 erzeugen mit runden Ecken
var DA_Rectangle1 = DA_Bibliothek.RoundRect(
    HoeheBzwBreiteInPixel,
    HoeheBzwBreiteInPixel,
    Dimension2,
    Dimension2
);

// ----- und mit Farbe füllen
DA_Rectangle1 = DA_Rectangle1.Fill(DA_StandardLinie, DA_FuellFarbe_Green);

// ----- Mittelpunkt der Animation des Rectangle 1 ermitteln
DA_Mittelpunkt = DA_Bibliothek.Translate2(Dimension1, -Dimension1);

// ----- 2D-Animationsachse des Rectangle 1 ermitteln bezüglich periodischer Elongation
//          Rectangle 1 wird entlang der Achse animiert
var AnimationsAchse = "Y"; // "Y" oder "X"
var DA_2DAnimationsAchse = eval("DA_Bibliothek." + AnimationsAchse + "Shear2Anim(DA_Elongation)");

// ----- 2D-Komposition aus Mittelpunkt und Animationsachse
DA_2DKomposition = DA_Bibliothek.Compose2(DA_Mittelpunkt, DA_2DAnimationsAchse);

// ----- Rectangle 1 mit 2D-Animationsachse verbinden
DA_Rectangle1 = DA_Rectangle1.Transform(DA_2DKomposition);

// +++++ 2D-Objekt Rectangle 2 +++++
//          Animation erfolgt entlang einer Achse (keine Rotation um die Achse)

// ----- Rectangle 2 erzeugen mit un-runden Ecken
var DA_Rectangle2 = DA_Bibliothek.Rect(HoeheBzwBreiteInPixel, HoeheBzwBreiteInPixel);

// ----- und füllen mit Farbe
DA_Rectangle2 = DA_Rectangle2.Fill(DA_StandardLinie, DA_FuellFarbe_Blue);

// ----- Mittelpunkt der Animation des Rectangle 2 ermitteln
DA_Mittelpunkt = DA_Bibliothek.Translate2(-Dimension1, -Dimension1);

// ----- 2D-Animationsachse des Rectangle 2 ermitteln bezüglich periodischer Elongation
// - - - Schritt 1
var DA_Zahl = ScriptWert_Nach_DA_Wert_Konvertieren(HoeheBzwBreiteInPixel / 4);
var DA_Produkt = DA_Multiplikation(DA_Zahl, DA_Elongation);

// - - - Schritt 2
var AbweichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation = 0;
var DA_AbschweichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation =
    ScriptWertWert_Nach_DA_Wert_Konvertieren(
        ichtungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation
    );

// - - - Schritt 3
DA_2DAnimationsAchse = DA_Bibliothek.Translate2Anim(
    DA_AbschweichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation,
    DA_Produkt
);

// ----- 2D-Komposition aus Mittelpunkt und 2D-Animationsachse
//          Mittelpunkt animiert entlang der 2D-Animationsachse
DA_2DKomposition = DA_Bibliothek.Compose2(DA_Mittelpunkt, DA_2DAnimationsAchse);

// ----- Rectangle 2 mit der 2D-Komposition verbinden
DA_Rectangle2 = DA_Rectangle2.Transform(DA_2DKomposition);

```



```

// +++++ 2D-Objekt Kreisstück +++++
// Animation erfolgt auf Kreisbahn
// ----- Kreisstück erzeugen
var DA_Kreisstueck = DA_Bibliothek.PieDegrees(-60, -120, HoeheBzwBreiteInPixel, HoeheBzwBreiteInPixel);

// ----- und mit Farbe füllen
DA_Kreisstueck = DA_Kreisstueck.Fill(DA_StandardLinie, DA_FuellFarbe_Purple);

// ----- Mittelpunkt der Animation des Kreisstückes ermitteln
DA_Mittelpunkt = DA_Bibliothek.Translate2(-Dimension1, Dimension1);

// ----- Rotation des Kreisstückes berechnen bezüglich periodischer Elongation
DA_2DRotation = DA_Bibliothek.Rotate2Anim(DA_Elongation);

// ----- 2D-Komposition aus Mittelpunkt und 2D-Rotation
DA_2DKomposition = DA_Bibliothek.Compose2(DA_Mittelpunkt, DA_2DRotation);

// ----- Kreisstück mit der 2D-Komposition verinden
DA_Kreisstueck = DA_Kreisstueck.Transform(DA_2DKomposition);

// ##### 2D-Animation erzeugen, die die 2D-Objekte in der Ebene animiert
// +++++ alle DA-Objekte als Feld
var DA_BildFeld = new Array(DA_Oval, DA_Rectangle1, DA_Rectangle2, DA_Kreisstueck);

// +++++ alle DA-Objekte überlagern zur kompletten Animation
var DA_2DAnimation = DA_Bibliothek.OverlayArray(DA_BildFeld);

// ##### 3D-Rotation der 2D-Animation erzeugen #####
// +++++ Achsen der Rotation ermitteln +++++
var Achse_X=0;
var Achse_Y=0;
var Achse_Z=0;

if (Achse_X_3DRotation)
{var Achse_X=1;}

if (Achse_Y_3DRotation)
{var Achse_Y=1;}

if (Achse_Z_3DRotation)
{var Achse_Z=1;}

// +++++ 3D-Vektor mit entsprechenden Achsen erzeugen
var DA_3DVektor = DA_Bibliothek.Vector3(Achse_X, Achse_Y, Achse_Z); // X,Y,Z
// wenn 1, so Rotation um diese Achse
// wenn 0, so keine Rotation um diese
// Achse

// +++++ rotierenden 3D-Vektor erzeugen +++++
var DA_3DVektor_Rotierend = DA_Bibliothek.Rotate3RateDegrees(DA_3DVektor, Geschwindigkeit_3DRotation);

// +++++ 3D-Rotation erzeugen +++++
var DA_3DRotation = DA_3DVektor_Rotierend.ParallelTransform2();

// ##### 2D-Animation und 3D-Rotation verbinden #####
var DA_2DAnimation_Mit3DRotation = DA_2DAnimation.Transform(DA_3DRotation);

// ##### Steuerung der 3D-rotierenden 2D-Animation erzeugen #####
// +++++ gesteuerte Animation erzeugen +++++
var DA_2DAnimation_Mit3DRotation_Gesteuert = new ActiveXObject("DirectAnimation.DAImage");

// +++++ Elemente der Steuerung erzeugen +++++
// Die Steuerung erfolgt über 2 verschachtelte Schleifen

// Schleife 1 (innerste Schleife)
// tue solange
// 2D-Animation im Raum rotieren lassen
// bis Druck der linken Maustaste erkannt wurde
// mit Schleifenende aktiviere die gesteuerte Rotation der 2D-Animation, welche mit der äußeren
// Schleife initialisiert wird, also erneut die Gesamtschleife aufrufen (Rekursion)
var DA_UntilSchleife = DA_Bibliothek.Until(
    DA_2DAnimation_Mit3DRotation,
    DA_LeftButtonDown,
    DA_2DAnimation_Mit3DRotation_Gesteuert

```



```

    );

    //      Schleife 2 (äußere Schleife)
    //      tue solange
    //      2D-Animation ohne Raumrotation
    //      bis Druck der linken Maustaste erkannt wurde
    //      mit Schleifenende aktiviere die innere Schleife also die Rotation der 2D-Animation
    DA_UntilSchleife = DA_Bibliothek.Until( DA_2DAnimation,
                                           DA_LeftButtonDown,
                                           DA_UntilSchleife
    );

    // +++++ Steuerung in die Animation einbauen +++++
    DA_2DAnimation_Mit3DRotation_Gesteuert.init(DA_UntilSchleife);

    // ##### DA-Control init #####
    DAControl.Image = DA_2DAnimation_Mit3DRotation_Gesteuert;

    // ##### Start der Animation #####
    DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT ID="DAControl"
            CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
            STYLE="position:relative; left:50; top:0;width:600;height:400"
    >
    </OBJECT>
    <BR>
    Schwingende <B>2D-Animation</B>
    <BR>
    im Wechsel mit
    <BR>
    2D-Animation <B>und</B> rotierende <B>3D-Animation</B>
    <BR>
    <BR>
    Klicke mit <B>linker Maustaste</B>, um zwischen den Animationen <B>zu wechseln</B>.
</BODY>
</HTML>

```

5.2.2.3.2. Sound

5.2.2.3.2.1. Sound ohne Kanalsteuerung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    function DirectAnimationStart()
    {
        // ##### allgemeine Angaben #####
        var SoundPfad = "";
        //      Leerkette zulässig
        //      auch .. kodierbar
        //      als Pfadtrenner muss \\ kodiert werden
        //      auch lokaler Pfad
        //      z.B. "file://c:\\d\\xm\\media\\"
        //      auch Internet-Url
        //      z.B. "http://www.test.de/test/mid"

        var SoundDatei = "sound.mid";

        // ##### DA-Bibliothek #####
        var DA_Bibliothek = DAControl.PixelLibrary;

        // ##### Sound erzeugen #####
        var SoundDateiMitPfad = SoundPfad + SoundDatei;

        // +++++ Sounddatei importieren
        var DA_Sound = DA_Bibliothek.ImportSound(SoundDateiMitPfad);

        // +++++ und implementieren
        DA_Sound = DA_Sound.Sound;

        // +++++ und als Endlos-Sound erzeugen
        DA_Sound = DA_Sound.Loop();
    }

```



```

// ##### DA-Init #####
DAControl.Sound = DA_Sound;

// ##### Start der Animation #####
DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
  <OBJECT ID="DAControl"
    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
    WIDTH=1 HEIGHT=1
  >
  </OBJECT>
</BODY>

```

5.2.2.3.2.2. Sound mit Kanalsteuerung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
  // ##### allgemeine Angaben #####
  var SoundPfad = "";
  // Leerkette zulässig
  // auch .. kodierbar
  // als Pfadtrenner muss \\ kodiert werden
  // auch lokaler Pfad
  // z.B. "file://c:\\dxm\\media\\"
  // auch Internet-Url
  // z.B. "http://www.test.de/test/mid"

  var SoundDatei = "sound.mid";

  var SoundDateiMitPfad = SoundPfad + SoundDatei;

  // ##### DA-Bibliothek #####
  var DA_Bibliothek = DAControl.PixelLibrary;

  // ##### diverse DA-Groessen #####
  var DA_Endlos_Timer = DA_Bibliothek.LocalTime;

  // ##### Sound erzeugen #####
  var SoundDateiMitPfad = SoundPfad + SoundDatei;

  // +++++ Sounddatei importieren
  var DA_Sound = DA_Bibliothek.ImportSound(SoundDateiMitPfad);

  // +++++ und implementieren
  DA_Sound = DA_Sound.Sound;

  // +++++ und als Endlos-Sound erzeugen
  DA_Sound = DA_Sound.Loop();

  // ##### DA-Sinus-Schwingung mit dynamischer Wertveränderung für periodischen Kanalwechsel
  var DA_SinusSchwingung = DA_Bibliothek.Sin(DA_Endlos_Timer);

  // ##### 2D-Rotation mit Sinus-Schwingung erzeugen #####
  var DA_2DRotation_Sinus = DA_Bibliothek.Rotate2Anim(DA_SinusSchwingung);

  // ##### Sound mit der 2D-Rotation verbinden #####
  DA_Sound = DA_Sound.PanAnim(DA_SinusSchwingung);

  // ##### DA-Init #####
  DAControl.Sound = DA_Sound;

  // ##### Start der Animation #####
  DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">

```




```

<OBJECT ID="DAControl"
        CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
        WIDTH=1 HEIGHT=1
>
</OBJECT>
<BR>
Stereo-Kanaele wandern links und rechts
</BODY>
</HTML>

```

5.2.2.3.3. Farbe

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
    // ##### allgemeine Angaben #####
    var TextDerAnimation          = "Das ist ein animierter DA-Text";
    var TextDerAnimation_FontArt  = "Arial"
    var TextDerAnimation_FontHoehe = 28;
    var TextDerAnimation_Spiegeln  = true;    // false für nicht spiegeln während der Animation
    var TextDer_Animation_Farbe_Rot_Anteil = 0.345; // >= 0
    var TextDer_Animation_Farbe_Gruen_Anteil = 0.6; // >= 0
    var TextDer_Animation_Farbe_Blau_Anteil = 0.5; // >= 0

    // ##### DA_Bibliothek #####
    var DA_Bibliothek = DAControl.PixelLibrary;

    // ##### diverse DA-Groessen #####
    var DA_EndlosTimer = DA_Bibliothek.LocalTime;
    var DA_Farbe_Blue = DA_Bibliothek.Blue;
    var DA_FuellFarbe_Blue = DA_Bibliothek.SolidColorImage(DA_Farbe_Blue);

    // ##### animierte Farbe mit fortlaufendem Farbwechsel erzeugen #####

    // +++++ RGB-Komponenten der Farbe erzeugen
    var DA_Farbe_Rot_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Rot_Anteil);
    var DA_Farbe_Gruen_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Gruen_Anteil);
    var DA_Farbe_Blau_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Blau_Anteil);

    // +++++ RGB-Komponenten mit Timer versehen, also automatische Wertveränderung der Komponente
    // es reicht aus, wenn 1 RGB-Komponente mit Timer versehen wird, da dann die
    // erzeugte Farbe bereits damit die automatische Wertveränderung bekommt
    DA_Farbe_Rot_Anteil = DA_Bibliothek.Mul(DA_Farbe_Rot_Anteil, DA_EndlosTimer);
    DA_Farbe_Gruen_Anteil = DA_Bibliothek.Mul(DA_Farbe_Gruen_Anteil, DA_EndlosTimer);
    DA_Farbe_Blau_Anteil = DA_Bibliothek.Mul(DA_Farbe_Blau_Anteil, DA_EndlosTimer);

    // +++++ und Farbe zusammenbauen
    DA_Farbe_Animiert = DA_Bibliothek.colorHslAnim(
        DA_Farbe_Rot_Anteil,
        DA_Farbe_Gruen_Anteil,
        DA_Farbe_Blau_Anteil
    );

    // ##### Font mit der animierten Farbe erzeugen #####
    DA_Font_Animiert = DA_Bibliothek.Font(
        TextDerAnimation_FontArt,
        TextDerAnimation_FontHoehe,
        DA_Farbe_Animiert
    );

    // ##### DA-Text mit animiertem Font erzeugen #####
    DA_Text = DA_Bibliothek.StringImage(TextDerAnimation, DA_Font_Animiert);

    // ##### DA-Init #####
    DAControl.Image = DA_Text;

    // nur bei IE3.x
    DAControl.BackgroundImage = DA_FuellFarbe_Blue;

    // ##### Start der Animation #####
    DAControl.Start();
}
-->
</SCRIPT>
</HEAD>

```



```

<BODY onload="DirectAnimationStart();">
  <OBJECT
    ID="DAControl"
    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
    STYLE="position:absolute; left:0; top:0; width:800; height:300"
  >
</OBJECT>
</BODY>
</HTML>

```

5.2.2.3.4. Text

5.2.2.3.4.1. Text ohne Hintegrund

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
    // ##### allgemeine Angaben #####
    var TextDerAnimation = "Das ist ein animierter DA-Text";
    var TextDerAnimation_FontArt = "Arial"
    var TextDerAnimation_FontHoehe = 28;
    var TextDerAnimation_Spiegeln = true; // false für nicht spiegeln während der Animation
    var TextDer_Animation_Farbe_Rot_Anteil = 0.345; // >= 0
    var TextDer_Animation_Farbe_Gruen_Anteil = 0.6; // >= 0
    var TextDer_Animation_Farbe_Blau_Anteil = 0.5; // >= 0

    // ##### DA_Bibliothek #####
    var DA_Bibliothek = DAControl.PixelLibrary;

    // ##### diverse DA-Groessen #####
    var DA_EndlosTimer = DA_Bibliothek.LocalTime;
    var DA_Farbe_Blue = DA_Bibliothek.Blue;
    var DA_FuellFarbe_Blue = DA_Bibliothek.SolidColorImage(DA_Farbe_Blue);

    // ##### animierte Farbe mit fortlaufendem Farbwechsel erzeugen #####

    // +++++ RGB-Komponenten der Farbe erzeugen
    var DA_Farbe_Rot_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Rot_Anteil);
    var DA_Farbe_Gruen_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Gruen_Anteil);
    var DA_Farbe_Blau_Anteil = DA_Bibliothek.DANumber(TextDer_Animation_Farbe_Blau_Anteil);

    // +++++ RGB-Komponenten mit Timer versehen, also automatische Wertveränderung der Komponente
    // es reicht aus, wenn 1 RGB-Komponente mit Timer versehen wird, da dann die
    // erzeugte Farbe bereits damit die automatische Wertveränderung bekommt
    DA_Farbe_Rot_Anteil = DA_Bibliothek.Mul(DA_Farbe_Rot_Anteil, DA_EndlosTimer);
    DA_Farbe_Gruen_Anteil = DA_Bibliothek.Mul(DA_Farbe_Gruen_Anteil, DA_EndlosTimer);
    DA_Farbe_Blau_Anteil = DA_Bibliothek.Mul(DA_Farbe_Blau_Anteil, DA_EndlosTimer);

    // +++++ und Farbe zusammenbauen
    DA_Farbe_Animiert = DA_Bibliothek.colorHslAnim(
        DA_Farbe_Rot_Anteil,
        DA_Farbe_Gruen_Anteil,
        DA_Farbe_Blau_Anteil
    );

    // ##### Font mit der animierten Farbe erzeugen #####
    DA_Font_Animiert = DA_Bibliothek.Font(
        TextDerAnimation_FontArt,
        TextDerAnimation_FontHoehe,
        DA_Farbe_Animiert
    );

    // ##### DA-Text mit animiertem Font erzeugen #####
    DA_Text = DA_Bibliothek.StringImage(TextDerAnimation, DA_Font_Animiert);

    // ##### DA-Init #####
    DAControl.Image = DA_Text;

    // nur bei IE3.x
    DAControl.BackgroundImage = DA_FuellFarbe_Blue;

    // ##### Start der Animation #####
    DAControl.Start();
}
-->
</SCRIPT>

```



```

</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
                    STYLE="position:absolute; left:0; top:0; width:800; height:300"
    >
    </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.4.2. Text mit Hintergrund

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    function DirectAnimationStart()
    {
        // ##### DA_Bibliothek #####
        var DA_Bibliothek = DAControl.PixelLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Farbe_Red      = DA_Bibliothek.ColorRgb(1, 0, 0);
        var DA_Farbe_Gray     = DA_Bibliothek.Gray;
        var DA_FuellFarbe_Gray = DA_Bibliothek.SolidColorImage(DA_Farbe_Gray);
        var DA_StandardFont   = DA_Bibliothek.DefaultFont;

        // ##### Font auf Basis Standard-Font #####
        // +++++ Fontfarbe festlegen
        var DA_Font = DA_StandardFont.Color(DA_Farbe_Red);

        // +++++ und Fonthöhe festlegen
        var FontHoehe = 48;
        DA_Font = DA_Font.Size(FontHoehe);

        // ##### DA-Text #####
        // +++++ Text als Script-String
        var Text="Das ist ein DA-Text"

        // +++++ Text mit Font verbinden
        var DA_Text = DA_Bibliothek.TextImage("Das ist ein DA-Text", DA_Font);

        // ##### DA-Bild der Animation aus Text und mit Farbe gefülltem Hintergrund
        // Hintergrund-Dimension laut OBJECT-TAG
        var DA_Bild = DA_Bibliothek.Overlay(DA_Text, DA_FuellFarbe_Gray);

        // ##### DA-Init #####
        DAControl.Image = DA_Bild;

        // ##### Start der Animation #####
        DAControl.Start()
    }
-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
                    STYLE="position:absolute; left:50; top:100; width:600; height:100"
    >
    </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.5. Font

5.2.2.3.5.1. Standardfont

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    function DirectAnimationStart()
    {
        // ##### allgemeine Angaben #####
        var AnzahlStellenNachDezimalKomma = 2;
        var FontHoehe_Animation_VonWert   = 64;

```



```

var FontHoehe_Animation_BisWert          = 128;
var FontHoehe_DauerEinerAnimationInSekunden = 3;

// ##### DA_Bibliothek #####
DA_Bibliothek = DAControl.PixelLibrary;

// ##### diverse DA-Groessen #####
var DA_Mausereignis_DruckLinkeTaste      = DA_Bibliothek.LeftButtonDown;
var DA_Farbe_Red                         = DA_Bibliothek.Red;
var DA_Farbe_Green                      = DA_Bibliothek.Green;
var StandardFont                        = DA_Bibliothek.DefaultFont;
var DA_Endlos_Timer                     = DA_Bibliothek.LocalTime;
var DA_Endlos_Timer_Als_DA_Kette        = DA_Endlos_Timer.toString(AnzahlStellenNachDezimalKomma);

// ##### DA-Text erzeugen #####
var DA_Text = new ActiveXObject("DirectAnimation.DAColor");

// ##### Steuerung der Text-Animation #####
//      Die Steuerung erfolgt über 2 verschachtelte Schleifen

//      Schleife 1 (innere Schleife)
//      tue solange
//          rote Farbe anzeigen
//      bis Druck der linken Maustaste erkannt wird
//      nach Schleifenende den Text animieren, der mit Schleife 2 initialisiert ist also Gesamtschleife
//      neu starten (Rekursion)
var DA_Schleife1 = DA_Bibliothek.Until(DA_Farbe_Red, DA_Mausereignis_DruckLinkeTaste, DA_Text);

//      Schleife 2 (äußere Schleife)
//      tue solange
//          Farbe animieren
//      bis Druck der linken Maustaste erkannt wird
//      nach Schleifenende rufe Schleife1 auf
var DA_Schleife2 = DA_Bibliothek.Until(DA_Farbe_Green, DA_Mausereignis_DruckLinkeTaste, DA_Schleife1);

// ##### DA-Text mit Steuerung versehen #####
var DA_Text_Gesteuert = DA_Text.Init(DA_Schleife2);

// ##### Sequenz eines Wertebereiches von Fonthöhen
// +++++ DA-Wertebereich-Folge erzeugen
var DA_FontHoeheWerteBereich = DA_Bibliothek.SlowInSlowOut(
    FontHoehe_Animation_VonWert,
    FontHoehe_Animation_BisWert,
    FontHoehe_DauerEinerAnimationInSekunden,
    0
);

// ---- und als endlos erzeugen
var DA_FontHoeheWerteBereich_EndlosFolge = DA_FontHoeheWerteBereich.RepeatForever();

// ##### DA-Font #####
//      Animation der Fonthöhe
// +++++ erzeugen
var DA_Font = StandardFont;

// +++++ mit gesteuertem DA-Text verbinden
DA_Font = DA_Font.Color(DA_Text_Gesteuert);

// +++++ Font mit der Sequenz der Fonthöhen verbinden
DA_Font = DA_Font.SizeAnim(DA_FontHoeheWerteBereich_EndlosFolge);

// ##### DA-Bild erzeugen #####
//      wird endlos animieren
var DA_Bild = DA_Bibliothek.TextImageAnim(DA_Endlos_Timer_Als_DA_Kette, DA_Font);

// ##### DA-Init #####
DAControl.Image = DA_Bild;

// ##### Start der Animation #####
DAControl.Start();
}

//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">

```



```

<OBJECT ID="DAControl"
        CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
        STYLE="left:100; top:100; width:400; height:300"
>
</OBJECT>
Klick in die Animation für Farbwechsel
</BODY>
</HTML>

```

5.2.2.3.5.2. Windows-Font

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
    // ##### allgemeine Angaben #####
    var WindowsFont_Name          = "Webdings";
    var WindowsFont_Groesse       = 35;
    var BewegungsGeschwindigkeitVerlangsamung = 2; // > 0, je höher um so langsamer
    var TastenKette               = "1234567:";
                                // Zeichen aus dem Font, die den Tasten mit Zeichen
                                // laut Kette entsprechen, also als ob man
                                // diese Tasten drücken würde

    var GleitRechtEck_BreiteFaktor = 0.016; // je höher um so breiter
    var GleitRechtEck_HoeheFaktor  = 0.016; // je höher um so höher

    // ##### DA_Bibliothek #####
    DA_Bibliothek = DAControl.MeterLibrary;

    // ##### diverse DA-Groessen #####
    var DA_Farbe_Black      = DA_Bibliothek.Black;
    var DA_Farbe_Blue       = DA_Bibliothek.Blue;
    var DA_FuellFarbe_Blue  = DA_Bibliothek.SolidColorImage(DA_Farbe_Blue);
    var DA_StandardLinie    = DA_Bibliothek.DefaultLineStyle;

    // ##### Windows-Font als DA-Objekt #####
    // +++++ Windows-Font als DA-Objekt für animierten Font
    var DA_InWindowsInstallierterFont_Animiert = DA_Bibliothek.Font(
                                                WindowsFont_Name,
                                                WindowsFont_Groesse,
                                                DA_Farbe_Black
                                                );

    // +++++ Windows-Font als DA-Objekt für nicht animierten Font
    var DA_InWindowsInstallierterFont_NichtAnimiert = DA_Bibliothek.Font(
                                                WindowsFont_Name,
                                                WindowsFont_Groesse,
                                                DA_Farbe_Blue
                                                );

    // alternativ auch kodierbar
    // var DA_InWindowsInstallierterFont_NichtAnimiert =
    //          DA_InWindowsInstallierterFont_Animiert.Color(DA_Farbe_Blue);
    //          es wird NUR die Farbe geändert gegenüber dem animierten Font

    // ##### DA-Bild animiert aus allen gewählten Zeichen aus dem Windows-Font
    // +++++ erzeugen
    var DA_AlleFontZeichenAlsGesamtBild_Animiert = DA_Bibliothek.StringImage(
                                                TastenKette,
                                                DA_InWindowsInstallierterFont_Animiert
                                                t
                                                );

    // +++++ und fixieren
    var DA_Punkt = DA_Bibliothek.Translate2(0,-.004); // (X, Y) als Abweichung von den Achsen
    DA_AlleFontZeichenAlsGesamtBild_Animiert =
        DA_AlleFontZeichenAlsGesamtBild_Animiert.Transform(DA_Punkt);

    // ##### DA-Bild nicht animiert aus allen gewählten Zeichen aus dem Windows-Font
    // +++++ erzeugen
    var DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert = DA_Bibliothek.StringImage(
                                                TastenKette,
                                                DA_InWindowsInstallierterFont_NichtAnimiert
                                                );

    // +++++ und fixieren
    DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert =

```



```

    t);
    DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert.Transform(DA_Punkt

// ##### Linie der 2D-Animation (Bewegung entlang der Linie)
// +++++ Start- und Endpunkt der Linie ermitteln
var DA_Punkt_BewegungStart = DA_Bibliothek.Point2(-.05,0); // (X, Y) als Abweichung von den Achsen
var DA_Punkt_BewegungEnde   = DA_Bibliothek.Point2(.05,0); // (X, Y) als Abweichung von den Achsen

// +++++ Linie als Pfad der Bewegung erzeugen
var DA_Linie = DA_Bibliothek.Line(DA_Punkt_BewegungStart,DA_Punkt_BewegungEnde);

// ##### Bewegung entlang der Linie #####
// +++++ erzeugen
var DA_Bewegung = DA_Bibliothek.FollowPath(DA_Linie, BewegungsGeschwindigkeitVerlangsamung);

// +++++ und endlos
var DA_Bewegung_Endlos = DA_Bewegung.RepeatForever();

// ##### 2D-Objekt Rechteck, das entlang der Linie sich bewegt
// +++++ erzeugen
var DA_Rectangle = DA_Bibliothek.Rect(GleitRechtEck_BreiteFaktor,GleitRechtEck_HoeheFaktor);

// +++++ und zeichnen
DA_Rectangle.Draw(DA_StandardLinie)

// +++++ und als beweglich erzeugen
var DA_Rectangle_Beweglich = DA_Rectangle.Transform(DA_Bewegung_Endlos);

// ##### DA-Bild der Ebene des Rechteckes, die sich entlang der Linie bewegt
// +++++ Koordinaten der Ebene ermitteln
var DA_Punkt_Beweglich_1 = DA_Bibliothek.Point2(-.008,-.008);
var DA_Punkt_Beweglich_2 = DA_Bibliothek.Point2(.008,.008);

// +++++ Ebene mit Bewegung entlang der Linie verbinden
DA_Punkt_Beweglich_1 = DA_Punkt_Beweglich_1.Transform(DA_Bewegung_Endlos);
DA_Punkt_Beweglich_2 = DA_Punkt_Beweglich_2.Transform(DA_Bewegung_Endlos);

// ##### Ebene mit animierten DA-Bild aus allen gewählten Zeichen aus dem Windows-Font verbinden
var DA_Bild_DerEbeneImRechteck_Beweglich = DA_AlleFontZeichenAlsGesamtBild_Animiert.Crop(
                                                                    DA_Punkt_Beweglich_1,
                                                                    DA_Punkt_Beweglich_2
                                                                    );

// ##### Animation zusammenbauen #####
// +++++ bewegliches Rechteck und bewegliche Ebene im Rechteck verbinden
var DA_Animation_Komplett = DA_Bibliothek.Overlay(
                                                                    DA_Rectangle_Beweglich,
                                                                    DA_Bild_DerEbeneImRechteck_Beweglich
                                                                    );

// +++++ dann nicht animiertes DA-Bild aus allen gewählten Zeichen aus dem Windows-Font einbinden
DA_Animation_Komplett = DA_Bibliothek.Overlay( DA_Animation_Komplett,
                                                                    DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert
                                                                    );

// alternativ auch kodierbar
//      var DA_Animation_Komplett_KomponentenFeld = new Array(
//      DA_Rectangle_Beweglich,
//      DA_Bild_DerEbeneImRechteck_Beweglich,
//      DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert
//      );
//      var DA_Animation_Komplett =
//      DA_Bibliothek.OverlayArray(DA_Animation_Komplett_KomponentenFeld);

// ##### DA-Init #####
DAControl.Image = DA_Animation_Komplett;

// nur für IE 3.x
DAControl.BackgroundImage = DA_FuellFarbe_Blue;

// ##### Start der Animation #####
DAControl.Start();
}
//-->
</SCRIPT>

```



```

</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
                    STYLE="position:absolute; left:20%; top:100%;width:500px;height:300"
    >
    </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.6. Grafik aus externer Bilddatei

5.2.2.3.6.1. Grafikfolge

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var BildDateiPfad = "";
    // Leerkette zulässig
    // auch .. kodierbar
    // als Pfadtrenner muss \\ kodiert werden
    // auch lokaler Pfad
    // z.B. "file://c:\\dxm\\media\\"
    // auch Internet-Url
    // z.B. "http://www.test.de/test/jpg/"

    var BildDatei1 = "bild.jpg";
    var BildDatei2 = "bild1.jpg";
    var BildDatei3 = "bild2.jpg";

    function DirectAnimationStart()
    {
        // ##### DA_Bibliothek #####
        DA_Bibliothek = DAControl.MeterLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Ereignis_Druck_Linke_Maustaste = DA_Bibliothek.LeftButtonDown;
        var DA_Ereignis_Druck_Rechte_Maustaste = DA_Bibliothek.RightButtonDown;
        var DA_Farbe_Black = DA_Bibliothek.Black;
        var DA_FuellFarbe_Black = DA_Bibliothek.SolidColorImage(DA_Farbe_Black)

        // ##### Bilder importieren #####
        var BildDatei1_MitPfad = BildDateiPfad + BildDatei1;
        var DA_Bild1 = DA_Bibliothek.ImportImage(BildDatei1_MitPfad);

        var BildDatei2_MitPfad = BildDateiPfad + BildDatei2;
        var DA_Bild2 = DA_Bibliothek.ImportImage(BildDatei2_MitPfad);

        var BildDatei3_MitPfad = BildDateiPfad + BildDatei3;
        var DA_Bild3 = DA_Bibliothek.ImportImage(BildDatei3_MitPfad);

        // ##### DA-Bild der Animation leer erzeugen #####
        var DA_BildDerAnimation = new ActiveXObject("DirectAnimation.DAImage");

        // ##### Steuerung der Animation erzeugen #####
        // anhand verschachtelter Schleifen
        // pro Bild wird eine Schleife erzeugt, die solange läuft, bis das Event Mausklick erkannt wurde
        // Eventerkennung durch Eventhandler per DA-Methode .AttachData()
        // die als Argument eine Referenz besitzt, die
        // mit Eintritt des Ereignisses aktiviert werden soll
        // Referenz muss DA-animiertes Objekt sein oder eine DA-Methode bzw. DA-Funktion

        // alle Bilder liegen in verschachtelten Schleifen, die sich gegenseitig aufrufen
        // das letzte zu animierende Bild muss in der innersten Schleife liegen
        // das erste zu animierende Bild muss in der äußersten Schleife liegen

        // Schleifenfolge:
        // die innerste Schleife aktiviert im Eventhandler die gesamte Animation,
        // wobei diese mit der äussersten Schleife initialisiert wird, also letztere
        // aktiviert mit Start der Animation (also endlose Schleifenfolge)
        // ansonsten muss eine Schleife im Eventhandler die nächst tiefer liegende Schleife
        // aktivieren
        // Schleifenfolge ist von innen nach außen aufzubauen

        // +++++ innerste Schleife erzeugen für Bild 3
        // tue solange

```




```

//          innerstes DA-Bild (letztes) animieren
//          bis Handler Ereignis erkannt hat
//          wobei der Handler dann DA_BildDerAnimation aktiviert, die mit der DA_Schleife_Aussen
//          initialisiert ist, also diese wieder startet
//          Event ist Druck der rechten Maustaste
var DA_Handler = DA_Ereignis_Druck_Rechte_Maustaste.AttachData(DA_BildDerAnimation);
var DA_Schleife_Innen = DA_Bibliothek.UntilEx(DA_Bild3, DA_Handler);

// +++++ mittlere Schleife erzeugen für Bild 2
//          tue solange
//          mittleres DA-Bild animieren
//          bis Handler Ereignis erkannt hat
//          wobei der Handler dann DA_Schleife_Innen aktiviert
//          Event ist Druck der linken Maustaste
var DA_Handler = DA_Ereignis_Druck_Linke_Maustaste.AttachData(DA_Schleife_Innen);
var DA_Schleife_Mitte = DA_Bibliothek.UntilEx(DA_Bild2, DA_Handler);

// +++++ äußere Schleife erzeugen für Bild 1
//          tue solange
//          äußeres DA-Bild animieren
//          bis Handler den Druck auf die linke Maustaste erkannt hat
//          wobei der Handler dann DA_Schleife_Mitte aktiviert
//          Event ist Druck der linken Maustaste
DA_Handler = DA_Ereignis_Druck_Linke_Maustaste.AttachData(DA_Schleife_Mitte);
var DA_Schleife_Aussen = DA_Bibliothek.UntilEx(DA_Bild1, DA_Handler);

// ##### DA-Bild der Animation mit der Steuerung versehen
DA_BildDerAnimation.Init(DA_Schleife_Aussen);

// ##### DA-Init #####
// nur bei IE 3.x
// DAControl.Image = DA_Bibliothek.Overlay( DA_BildDerAnimation, DA_FuellFarbe_Black);

// sonst
DAControl.Image = DA_BildDerAnimation;

// ##### Start der Animation #####
DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
                    WIDTH=300
                    HEIGHT=300
    >
    </OBJECT>
    <BR>
    <BR>
    Bildwechsel per Klick mit linker Maustaste
    <BR>
    und beim letzten Bild Sprung auf erstes Bild per Klick mit rechter Maustaste
</BODY>
</HTML>

```

5.2.2.3.6.2. **Grafik scrollend**

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var BildDateiPfad          = "";          // Leerkette zulässig
                                          // auch .. kodierbar
                                          // als Pfadtrenner muss \\ kodiert werden
                                          // auch lokaler Pfad
                                          //          z.B. "file://c:\\dxm\\media\\"
                                          //          auch Internet-Url
                                          //          z.B. "http://www.test.de/test/jpg/"

    var BildDatei              = "bild.jpg";
    var ScrollGeschwindigkeitFaktor_Vertikal = -10; // 0 so kein scrollen
                                          // > 0 so scrollen nach oben
                                          // < 0 so scrollen nach unten
                                          // je höher um so schneller

```



```

var ScrollGeschwindigkeitFaktor_Horizontal = -10; // 0 so kein scrollen
// > 0 so scrollen nach rechts
// < 0 so scrollen nach links
// je höher um so schneller

var Hintergrund_Rand_Oben_Bzw_Unten = 30; // Pixel
var Hintergrund_Rand_Links_Bzw_Rechts = 50; // Pixel

var Hintergrund_Breite_Minimal = 30; // Pixel
var Hintergrund_Hoehe_Minimal = 30; // Pixel

var Hintergrund_Bild_Kacheln = true; // true für kacheln, false für kein kacheln
// Achtung: nur kacheln erzeugt endloses Scrollen
// ohne kacheln verschwindet des Bild für immer

function Dimension_Des_DAControl_An_BODY_Dimension_Anpassen()
{
    if(document.body.clientWidth > Hintergrund_Rand_Links_Bzw_Rechts )
    {DAControl.style.width = document.body.clientWidth - Hintergrund_Rand_Links_Bzw_Rechts;}
    else
    {DAControl.style.width = Hintergrund_Breite_Minimal;}

    if(document.body.clientHeight > Hintergrund_Rand_Oben_Bzw_Unten)
    {DAControl.style.height = document.body.clientHeight - Hintergrund_Rand_Oben_Bzw_Unten;}
    else
    {DAControl.style.height = Hintergrund_Hoehe_Minimal;}
}

function DirectAnimationStart()
{
    var BildDateiMitPfad = BildDateiPfad + BildDatei;

    // ##### DA_Bibliothek #####
    DA_Bibliothek = DAControl.MeterLibrary;

    // ##### Dimension des Control an aktuelle Body-Dimension anpassen
    // Achtung: keine automatische Erkennung von resize
    Dimension_Des_DAControl_An_BODY_Dimension_Anpassen();

    // ##### Scrollgeschwindigkeiten ermitteln #####
    // +++++ DA-Scrollgeschwindigkeit vertikal
    // ----- Faktor erzeugen
    ScrollGeschwindigkeitFaktor_Vertikal /= 1000;
    var DA_ScrollGeschwindigkeitFaktor_Vertikal =
        DA_Bibliothek.DANumber(ScrollGeschwindigkeitFaktor_Vertikal);

    // ----- sich selbständig fortlaufend um 1 erhöhendern Zähler erzeugen
    var DA_ScrollGeschwindigkeit_Vertikal = DA_Bibliothek.Integral(DA_ScrollGeschwindigkeitFaktor_Vertikal);

    // +++++ DA-Scrollgeschwindigkeit horizontal ermitteln
    // ----- Faktor erzeugen

    ScrollGeschwindigkeitFaktor_Horizontal /= 1000;
    var DA_ScrollGeschwindigkeitFaktor_Horizontal =
        DA_Bibliothek.DANumber(ScrollGeschwindigkeitFaktor_Horizontal);

    // ----- sich selbständig fortlaufend um 1 erhöhendern Zähler erzeugen
    var DA_ScrollGeschwindigkeit_Horizontal = DA_Bibliothek.Integral(DA_ScrollGeschwindigkeitFaktor_Horizontal);

    // +++++ gesamte DA-Scrollgeschwindigkeit der Animation ermitteln
    var DA_ScrollGeschwindigkeit_Gesamt = DA_Bibliothek.Translate2Anim(
        DA_ScrollGeschwindigkeit_Horizontal,
        DA_ScrollGeschwindigkeit_Vertikal
    );

    // ##### DA-Bild der Grafik #####
    // +++++ Bilddatei importieren
    var DA_Bild = DA_Bibliothek.ImportImage(BildDateiMitPfad);

    // +++++ und kacheln
    if (Hintergrund_Bild_Kacheln)
    {DA_Bild = DA_Bild.Tile();}

    // +++++ und mit Scrollen verbinden

```



```

        DA_Bild = DA_Bild.Transform(DA_ScrollGeschwindigkeit_Gesamt);

        // ##### DA-Init #####
        DAControl.Image = DA_Bild;

        // ##### Animation starten #####
        DAControl.Start();
    }
//-->
</SCRIPT>
</HEAD>

<BODY onload="DirectAnimationStart();">
    <OBJECT ID="DAControl"
            CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
            STYLE="position:absolute; left:50; top:25; z-index:-1" <!-- z-index ist wichtig -->
            >

    </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.6.3. Grafik rotierend

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var BildDateiPfad = "";
                                // Leerkette zulässig
                                // auch .. kodierbar
                                // als Pfadtrenner muss \\ kodiert werden
                                // auch lokaler Pfad
                                // z.B. "file://c:\\dxm\\media\\"
                                // auch Internet-Url
                                // z.B. "http://www.test.de/test/jpg/"

    var BildDatei = "bild.jpg";
    var RotationsGeschwindigkeit = 5;

    function DirectAnimationStarten()
    {
        // ##### DA_Bibliothek #####
        DA_Bibliothek = DAControl.MeterLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Farbe_White = DA_Bibliothek.White;
        var DA_FuellFarbe_White = DA_Bibliothek.SolidColorImage(DA_Farbe_White);

        // ##### 2D-Rotation endlos erzeugen #####
        var DA_Rotation = DA_Bibliothek.Rotate2Rate(RotationsGeschwindigkeit);

        // oder auch z.B. 60 Grad-Drehung pro Sekunde
        // var AnzahlGrade=60;
        // var DA_Rotation = DA_Bibliothek.Rotate2RateDegrees(AnzahlGrade);

        // ##### Bilddatei importieren #####
        var DateinameMitPfad = BildDateiPfad + BildDatei;
        var DA_Bild = DA_Bibliothek.ImportImage(DateinameMitPfad);

        // ##### DA-Bild mit der Rotation verbinden #####
        DA_Bild = DA_Bild.Transform(DA_Rotation);

        // ##### DA-Init #####
        DAControl.Image = DA_Bild;

        // nur bei IE 3.x
        DAControl.BackgroundImage = DA_FuellFarbe_White;

        // ##### Start der Animation #####
        DAControl.Start();
    }
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStarten();">
    <OBJECT ID="DAControl"
            CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"

```



```

        >
        STYLE="position:absolute ;width:200;height:200"
    </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.7. Sequenz

5.2.2.3.7.1. Sequenz mit zeitlicher Begrenzung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    function DirectAnimationStart()
    {
        // ##### allgemeine Angaben #####
        var TextFeld          = new Array("Text_5Sekunden", "Text_7Sekunden", "Text_3Sekunden", "Ende");
        Text_AnzeigeDauerFeld = new Array(5,7,3, 1);          // in Sekunden
                                                                // letztes Feldelement ist wertmäßig egal

        var FontHoehe        = 16;

        // ##### DA_Bibliothek #####
        DA_Bibliothek = DAControl.PixelLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Farbe_Red      = DA_Bibliothek.Red;
        var DA_Farbe_Blue     = DA_Bibliothek.Blue;
        var DA_Standard_Font  = DA_Bibliothek.DefaultFont;

        // ##### Zeichenkettensequenz #####
        // +++++ leer erzeugen
        var DA_ZeichenKettenSequenz = DA_Bibliothek.DAString("");

        // +++++ und ohne Animationsdauer
        DA_ZeichenKettenSequenz = DA_ZeichenKettenSequenz.Duration(0);

        // +++++ dann zusammenbauen aus den einzelnen Texten
        var TextFeld_Laenge = TextFeld.length;
        for (var i=0; i < TextFeld_Laenge; i++)
        {
            var DA_Zeichenkette          = DA_Bibliothek.DAString(TextFeld[i]);
            DA_Zeichenkette               = DA_Zeichenkette.Duration(Text_AnzeigeDauerFeld[i]);
            DA_ZeichenKettenSequenz      = DA_Bibliothek.Sequence(DA_ZeichenKettenSequenz,
                                                                    DA_Zeichenkette);
        }

        // ##### Farbsequenz #####
        // +++++ Komponenten der Sequenz erzeugen
        var FarbAnzeigeDauer            = 1;          // in Sekunden
        var DA_Farbe_Red_MitAnzeigeDauer = DA_Farbe_Red.Duration(FarbAnzeigeDauer);
        var DA_Farbe_Blue_MitAnzeigeDauer = DA_Farbe_Blue.Duration(FarbAnzeigeDauer);

        // +++++ Sequenz erzeugen aus den Komponenten
        var DA_FarbeSequenz = DA_Bibliothek.Sequence( DA_Farbe_Red_MitAnzeigeDauer,
                                                       DA_Farbe_Blue_MitAnzeigeDauer
                                                       );

        // +++++ und endlos damit auch Farbwechsel sichtbar ist
        DA_FarbeSequenz = DA_FarbeSequenz.RepeatForever();

        // ##### Font-Animation anhand Sequenz #####
        // +++++ erzeugen mit Farbsequenz
        var DA_Font = DA_Standard_Font.Color(DA_FarbeSequenz);

        // +++++ Fonthöhe erzeugen
        DA_Font = DA_Font.Size(16);

        // ##### Animation erzeugen #####
        // +++++ aus der Kettensequenz erzeugen mit dem Font
        var DA_Bild = DA_Bibliothek.StringImageAnim(DA_ZeichenKettenSequenz, DA_Font);

        // +++++ Position der Animation erzeugen
        var DA_2DPunkt = DA_Bibliothek.Translate2(0, 130)

```



```

// +++++ und DA-Bild positionieren
DA_Bild = DA_Bild.Transform(DA_2DPunkt);

// ##### DA-Init #####
DAControl.Image = DA_Bild;

// ##### Animation starten #####
DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
  <OBJECT ID="DAControl"
    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
    WIDTH=450 HEIGHT=300 ALIGN=LEFT
  >
  </OBJECT>
</BODY>
</HTML>

```

5.2.2.3.7.2. Sequenz mit Wertbereich-Begrenzung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
function DirectAnimationStart()
{
  // ##### allgemeine Angaben #####
  var AnzahlStellenNachDezimalKomma = 2;
  var FontHoehe_Animation_VonWert = 64;
  var FontHoehe_Animation_BisWert = 128;
  var FontHoehe_DauerEinerAnimationInSekunden = 3;

  // ##### DA_Bibliothek #####
  DA_Bibliothek = DAControl.PixelLibrary;

  // ##### diverse DA-Groessen #####
  var DA_Mausereignis_DruckLinkeTaste = DA_Bibliothek.LeftButtonDown;
  var DA_Farbe_Red = DA_Bibliothek.Red;
  var DA_Farbe_Green = DA_Bibliothek.Green;
  var StandardFont = DA_Bibliothek.DefaultFont;
  var DA_Endlos_Timer = DA_Bibliothek.LocalTime;
  var DA_Endlos_Timer_Als_DA_Kette = DA_Endlos_Timer.toString(AnzahlStellenNachDezimalKomma);

  // ##### DA-Text erzeugen #####
  var DA_Text = new ActiveXObject("DirectAnimation.DAColor");

  // ##### Steuerung der Text-Animation #####
  // Die Steuerung erfolgt über 2 verschachtelte Schleifen

  // Schleife 1 (innere Schleife)
  // tue solange
  // rote Farbe anzeigen
  // bis Druck der linken Maustaste erkannt wird
  // nach Schleifenende den Text animieren, der mit Schleife 2 initialisiert ist also Gesamtschleife
  // neu starten (Rekursion)
  var DA_Schleife1 = DA_Bibliothek.Until(DA_Farbe_Red, DA_Mausereignis_DruckLinkeTaste, DA_Text);

  // Schleife 2 (äußere Schleife)
  // tue solange
  // Farbe animieren
  // bis Druck der linken Maustaste erkannt wird
  // nach Schleifenende rufe Schleife1 auf
  var DA_Schleife2 = DA_Bibliothek.Until(DA_Farbe_Green, DA_Mausereignis_DruckLinkeTaste, DA_Schleife1);

  // ##### DA-Text mit Steuerung versehen #####
  var DA_Text_Gesteuert = DA_Text.Init(DA_Schleife2);

  // ##### Sequenz eines Wertbereiches von Fonthöhen
  // +++++ DA-Wertebereich-Folge erzeugen
  var DA_FontHoeheWertebereich = DA_Bibliothek.SlowInSlowOut(
    FontHoehe_Animation_VonWert,
    FontHoehe_Animation_BisWert,
    FontHoehe_DauerEinerAnimationInSekunden,

```



```

0
);

// ----- und als endlos erzeugen
var DA_FontHoeheWerteBereich_EndlosFolge = DA_FontHoeheWerteBereich.RepeatForever();

// ##### DA-Font #####
// Animation der Fonthöhe
// +++++ erzeugen
var DA_Font = StandardFont;

// +++++ mit gesteuertem DA-Text verbinden
DA_Font = DA_Font.Color(DA_Text_Gesteuert);

// +++++ Font mit der Sequenz der Fonthöhen verbinden
DA_Font = DA_Font.SizeAnim(DA_FontHoeheWerteBereich_EndlosFolge);

// ##### DA-Bild erzeugen #####
// wird endlos animieren
var DA_Bild = DA_Bibliothek.TextImageAnim(DA_Endlos_Timer_Als_DA_Kette, DA_Font);

// ##### DA-Init #####
DAControl.Image = DA_Bild;

// ##### Start der Animation #####
DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT ID="DAControl"
            CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
            STYLE="left:100; top:100; width:400; height:300"
    >
    </OBJECT>
    Klick in die Animation für Farbwechsel
</BODY>
</HTML>

```

5.2.2.3.8. Event

5.2.2.3.8.1. Eventhandler ohne Erweiterung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var BildDateiPfad = "";

    // Leerkette zulässig
    // auch .. kodierbar
    // als Pfadtrenner muss \\ kodiert werden
    // auch lokaler Pfad
    // z.B. "file://c:\\dxm\\media\\"
    // auch Internet-Url
    // z.B. "http://www.test.de/test/jpg/"

    var BildDatei1 = "bild.jpg";
    var BildDatei2 = "bild1.jpg";
    var BildDatei3 = "bild2.jpg";

    function DirectAnimationStart()
    {
        // ##### DA_Bibliothek #####
        DA_Bibliothek = DAControl.MeterLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Ereignis_Druck_Linke_Maustaste = DA_Bibliothek.LeftButtonDown;
        var DA_Ereignis_Druck_Rechte_Maustaste = DA_Bibliothek.RightButtonDown;
        var DA_Farbe_Black = DA_Bibliothek.Black;
        var DA_FuellFarbe_Black = DA_Bibliothek.SolidColorImage(DA_Farbe_Black);

        // ##### Bilder importieren #####
        var BildDatei1_MitPfad = BildDateiPfad + BildDatei1;
        var DA_Bild1 = DA_Bibliothek.ImportImage(BildDatei1_MitPfad);

        var BildDatei2_MitPfad = BildDateiPfad + BildDatei2;
        var DA_Bild2 = DA_Bibliothek.ImportImage(BildDatei2_MitPfad);
    }

```



```

var BildDatei3_MitPfad          = BildDateiPfad + BildDatei3;
var DA_Bild3                    = DA_Bibliothek.ImportImage(BildDatei3_MitPfad);

// ##### DA-Bild der Animation leer erzeugen #####
var DA_BildDerAnimation= new ActiveXObject("DirectAnimation.DAImage");

// ##### Steuerung der Animation erzeugen #####
//          anhand verschachtelter Schleifen
//          pro Bild wird eine Schleife erzeugt, die solange läuft, bis das Event Mausklick erkannt wurde
//          Eventerkennung durch Eventhandler per DA-Methode .AttachData()
//          die als Argument eine Referenz besitzt, die
//          mit Eintritt des Ereignisses aktiviert werden soll
//          Referenz muss DA-animiertes Objekt sein oder eine DA-Methode bzw. DA-Funktion

//          alle Bilder liegen in verschachtelten Schleifen, die sich gegenseitig aufrufen
//          das letzte zu animierende Bild muss in der innersten Schleife liegen
//          das erste zu animierende Bild muss in der äußersten Schleife liegen

//          Schleifenfolge:
//          die innerste Schleife aktiviert im Eventhandler die gesamte Animation,
//          wobei diese mit der äussersten Schleife initialisiert wird, also letztere
//          aktiviert mit Start der Animation (also endlose Schleifenfolge)
//          ansonsten muss eine Schleife im Eventhandler die nächst tiefer liegende Schleife
//          aktivieren
//          Schleifenfolge ist von innen nach außen aufzubauen

// +++++ innerste Schleife erzeugen für Bild 3
//          tue solange
//          innerstes DA-Bild (letztes) animieren
//          bis Handler Ereignis erkannt hat
//          wobei der Handler dann DA_BildDerAnimation aktiviert, die mit der DA_Schleife_Aussen
//          initialisiert ist, also diese wieder startet
//          Event ist Druck der rechten Maustaste
var DA_Handler = DA_Ereignis_Druck_Rechte_Maustaste.AttachData(DA_BildDerAnimation);
var DA_Schleife_Innen = DA_Bibliothek.UntilEx(DA_Bild3, DA_Handler);

// +++++ mittlere Schleife erzeugen für Bild 2
//          tue solange
//          mittleres DA-Bild animieren
//          bis Handler Ereignis erkannt hat
//          wobei der Handler dann DA_Schleife_Innen aktiviert
//          Event ist Druck der linken Maustaste
var DA_Handler = DA_Ereignis_Druck_Linke_Maustaste.AttachData(DA_Schleife_Innen);
var DA_Schleife_Mitte = DA_Bibliothek.UntilEx(DA_Bild2, DA_Handler);

// +++++ äußere Schleife erzeugen für Bild 1
//          tue solange
//          äußeres DA-Bild animieren
//          bis Handler den Druck auf die linke Maustaste erkannt hat
//          wobei der Handler dann DA_Schleife_Mitte aktiviert
//          Event ist Druck der linken Maustaste
DA_Handler = DA_Ereignis_Druck_Linke_Maustaste.AttachData(DA_Schleife_Mitte);
var DA_Schleife_Aussen = DA_Bibliothek.UntilEx(DA_Bild1, DA_Handler);

// ##### DA-Bild der Animation mit der Steuerung versehen
DA_BildDerAnimation.Init(DA_Schleife_Aussen);

// ##### DA-Init #####
// nur bei IE 3.x
// DAControl.Image = DA_Bibliothek.Overlay( DA_BildDerAnimation, DA_FuellFarbe_Black);

// sonst
DAControl.Image = DA_BildDerAnimation;

// ##### Start der Animation #####
DAControl.Start();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
  <OBJECT          ID="DAControl"
                  CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"

```




```

        WIDTH=300
        HEIGHT=300
    >
</OBJECT>
<BR>
<BR>
Bildwechsel per Klick mit linker Maustaste
<BR>
und beim letzten Bild Sprung auf erstes Bild per Klick mit rechter Maustaste
</BODY>
</HTML>

```

5.2.2.3.8.2. Eventhandler mit Erweiterung

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    // ##### globale Variablen #####
    var DA_Bibliothek;
    var DA_Event;
    var DA_Zahl_DieImEventVerarbeitetWird;
    var EventIgnorieren;

    // ##### Event registrieren #####
    //      aktiviert laut onclick im INPUT-Tag
    function DA_Event_Registrieren()
    {
        // DA-Zahl mit beliebigem DA-Wert als Dummy-Parameter für korrekte Funktionsweise von .TriggerEvent()
        var DA_Zahl_MitBeliebigenWert = DA_Bibliothek.DANumber(0);

        // jedes Event registrieren, wobei Event-Typ egal ist
        DA_Bibliothek.TriggerEvent( DA_Event, DA_Zahl_MitBeliebigenWert );
    }

    // ##### Konvertierung DA-Wert nach Script-Wert #####
    function DA_Zahl_MitWert_Nach_ScriptWert_Konvertieren(DA_Zahl)
    {return DA_Zahl.Extract();}

    // ##### Reaktion auf Event ausführen #####
    function Auf_DA_Event_Reagieren() // Funktionsbezeichner wird im Quellcode auch als Kette verwendet !!
    {
        // prüfen ob Event ignoriert werden muss (EventIgnorieren muss global sein)
        if (EventIgnorieren)
        {
            // Event ignorieren aber nächste Eventverarbeitung freigeben
            EventIgnorieren = false;
        }
        else
        {
            // Event bearbeiten

            // +++++ DA_Zahl_DieImEventVerarbeitetWird importieren
            var ZeigerAuf_DA_Zahl = DA_Zahl_DieImEventVerarbeitetWird;

            // +++++ Wert des DA_Zahl als Script-Wert erzeugen
            var DatenZumEvent_AlsScriptWert =

            DA_Zahl_MitWert_Nach_ScriptWert_Konvertieren(ZeigerAuf_DA_Zahl);

            // +++++ Meldung auf Bildschirm, wobei alert eine Null am Ende des Wertes abschneidet
            //      und intern toString() verwendet
            alert(
                + DatenZumEvent_AlsScriptWert
                + "\n\nNull am Ende wird nicht angezeigt!"
            );

            // +++++ nächste Eventverarbeitung ingnorieren
            EventIgnorieren = true;
        }
    }

    // ##### Direct Animation verwalten #####
    function DirectAnimationStart()
    {
        function StandardFont_Verandern_UndZeigerLiefern(DA_Farbe,FontHoehe)

```



```

    {
        var DA_Font = DA_StandardFont.Color(DA_Farbe);
        DA_Font = DA_Font.Size(FontHoehe);

        return DA_Font;
    }

    // ##### DA_Bibliothek #####
    DA_Bibliothek = DAControl.MeterLibrary;

    // ##### diverse DA-Groessen #####
    var DA_Farbe_Blue           = DA_Bibliothek.Blue;
    var DA_FuellFarbe_Blue     = DA_Bibliothek.SolidColorImage(DA_Bibliothek.Blue);
    var DA_Farbe_Yellow        = DA_Bibliothek.Yellow;
    var DA_StandardFont        = DA_Bibliothek.defaultFont;
    var DA_ZahlMitWert_1       = DA_Bibliothek.DANumber(1);

    DA_Event                    = DA_Bibliothek.AppTriggeredEvent();
                                // allgemeines Event-Objekt (muss globale Variable sein)
    DA_Zahl_DieImEventVerarbeitetWird = DA_ZahlMitWert_1;           // globale Variable

    // ##### Eventhandler mit der Funktion "Auf_DA_Event_Reagieren" verbinden
    //          um diese im Eventfall aufrufen zu können, der eintritt, wenn die Überwachung
    //          des onclick-Ereignisses per Funktion DA_Event_Registrieren()
    //          ein durch User getätigtes onclick erkannt hat
    var DA_Event_MitHandler = DA_Event.NotifyScript( "Auf_DA_Event_Reagieren" );
                                // Funktion "Auf_DA_Event_Reagieren" kann keine Argument besitzen
                                //          und muss ein zu verarbeitendes DA-Objekt importieren

    // ##### im erweiterten Eventhandler zu behandelndes DA-Objekt als DA-Kette erzeugen
    var DA_Zahl_DieImEventVerarbeitetWird_als_DA_String = DA_Zahl_DieImEventVerarbeitetWird.toString(3);

    // ##### Animationsbild erzeugen #####
    // +++++ DA-Font erzeugen für String auf Basis des DA-Standard-Fonts
    var DA_StandardFont_Veraendert = StandardFont_Verändern_UndZeigerLiefern(DA_Farbe_Yellow, -20);

    // +++++ im erweiterten Eventhandler zu behandelndes DA-Objekt mit DA-Font als Bild erzeugen
    var DA_StringAlsBild = DA_Bibliothek.StringImageAnim(DA_Zahl_DieImEventVerarbeitetWird_als_DA_String,
                                                         DA_StandardFont_Veraendert
                                                         );

    // +++++ Bild mit Hintergrundfarbe versehen per Überlagerung in der Dimension des Hintergrund laut OBJECT-Tag
    //          (width und heigth)
    var DA_Bild = DA_Bibliothek.Overlay(DA_StringAlsBild, DA_FuellFarbe_Blue);

    // ##### DA-Init #####
    // +++++ und Eventsteuerung als Verhaltensweise der Animation zuweisen
    DAControl.AddBehaviorToRun(DA_Event_MitHandler);
    //          nächste Eventverarbeitung zulassen
    EventIgnorieren = false;

    DAControl.image = DA_Bild;

    // ##### Start der Animation #####
    DAControl.start();
}

-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT          ID="DAControl"
                    CLASSID="CLSID:69AD90EF-1C20-11d1-8801-00C04FC29D46"
                    STYLE="position:absolute; width:300; height:80; top:85px; left:100px"

    >
    </OBJECT>
    <BR>
    <INPUT  TYPE=button
          NAME="ID_InputButton"
          VALUE="DA-Zahl anzeigen"
          onclick="DA_Event_Registrieren();">

</BODY>
</HTML>

```



5.2.2.3.8.3. Eventfähigkeit eines DA-Objektes

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var NeueUrl = "";                // Url des DA-Links

    var IMGPFad = "";                // Leerkette zulässig
                                    // auch .. kodierbar
                                    // als Pfadtrenner muss \\ kodiert werden
                                    // auch lokaler Pfad
                                    // z.B. "file://c:\\dxm\\media\\"
                                    // auch Internet-Url
                                    // z.B. "http://www.test.de/test/jpg/"

    var SoundPfad = "";              // Leerkette zulässig
                                    // auch .. kodierbar
                                    // als Pfadtrenner muss \\ kodiert werden
                                    // auch lokaler Pfad
                                    // z.B. "file://c:\\dxm\\media\\"
                                    // auch Internet-Url
                                    // z.B. "http://www.test.de/test/mid/"

    var SoundDatei = "sound.mid"
    var BildDatei = "bild.jpg"

    var Text = "Klicke auf Button für Wechsel nach " + NeueUrl;
    var Text_FontArt = "Arial"
    var Text_FontHoehe = 12;

    function UrlAusfuehren()
    { window.top.location.href = NeueUrl; }

    function DirectAnimationStart()
    {
        var SoundDateiMitPfad = SoundPfad + SoundDatei;
        var BildDateiMitPfad = IMGPFad + BildDatei;

        // ##### DA-Bibliothek #####
        var DA_Bibliothek = DAControl.MeterLibrary;

        // ##### diverse DA-Groessen #####
        var DA_Farbe_White = DA_Bibliothek.White;
        var DA_Farbe_Black = DA_Bibliothek.Black;
        var DA_FuellFarbe_Black = DA_Bibliothek.SolidColorImage(DA_Farbe_Black);
        var DA_Farbe_Red = DA_Bibliothek.Red;
        var DA_FuellFarbe_Red = DA_Bibliothek.SolidColorImage(DA_Farbe_Red);
        var DA_Farbe_Blue = DA_Bibliothek.Blue;
        var DA_FuellFarbe_Blue = DA_Bibliothek.SolidColorImage(DA_Farbe_Blue);
        var DA_Event_Druck_LinkerMaustaste = DA_Bibliothek.LeftButtonDown;
        var DA_Sound_Off = DA_Bibliothek.Silence;
        var DA_Leeres_DA_Bild = DA_Bibliothek.EmptyImage;

        // ##### DA-Sound #####
        // +++++ DA-Sound-Datei importieren
        var DA_Sound = DA_Bibliothek.ImportSound(SoundDateiMitPfad);

        // ----- erzeugen
        DA_Sound = DA_Sound.Sound;

        // ----- und endlos
        DA_Sound = DA_Sound.Loop();

        // ##### DA-Text als Link-Text #####
        // +++++ DA-Font
        // ----- erzeugen
        var DA_Font = DA_Bibliothek.Font(Text_FontArt, Text_FontHoehe, DA_Farbe_White);
        // ----- und Fettschrift
        DA_Font = DA_Font.Bold();

        // +++++ DA-Text mit Font erzeugen
        var DA_Text = DA_Bibliothek.StringImage(Text, DA_Font);

        // ##### Text-Box auf Basis des DA-Textes #####

```



```

// +++++ erzeugen
var DA_TextBox = DA_Text.BoundingBox;

// +++++ Dimension ermitteln
var DA_TextBox_Minimum = DA_TextBox.Min;
var DA_TextBox_Maximum = DA_TextBox.Max;

// +++++ mit Hintergrundfarbe füllen
var DA_TextBox_MitHintergrund = DA_FuellFarbe_Black.Crop(DA_TextBox_Minimum,DA_TextBox_Maximum);

// ##### DA-Text mit Textbox verbinden #####
var DA_Text_Animation = DA_Bibliothek.Overlay(DA_Text,DA_TextBox_MitHintergrund);

// ##### DA-Rectangle erzeugen als Klick-Button für Url-Wechsel #####
// +++++ 2D-Koordinaten des Button erzeugen, also bezüglich DA-Koordinatensystem
var DA_2DPunkt1 = DA_Bibliothek.Point2(-0.006,-0.006); // Abweichung vom Ursprung des Koordinatensystemes
var DA_2DPunkt2 = DA_Bibliothek.Point2( 0.006, 0.006); // Abweichung vom Ursprung des Koordinatensystemes

// +++++ Rectangle erzeugen und innerhalb der Dimension mit Farbe füllen
var DA_Rectangle = DA_FuellFarbe_Red.Crop(DA_2DPunkt1,DA_2DPunkt2);

// +++++ Lage des DA-Button als Rectangle erzeugen
// ----- Lage bezüglich X-Achse
var Wert_X_Achse = 0; // X, positiv so nach rechts, negativ so nach links, 0 ist zentriert
var DA_Wert_X_Achse = DA_Bibliothek.DANumber(Wert_X_Achse);
// Hinweis: Umrechnung in Pixeleinheit möglich per
// DA_Wert_X_Achse = DA_Bibliothek.Mul( DA_Wert_X_Achse,
// DA_Bibliothek.Pixel
// );

// ----- Lage bezüglich Y-Achse
var Wert_Y_Achse = -.019; // Y, positiv so nach oben, negativ so nach unten, 0 ist zentriert
var DA_Wert_Y_Achse = DA_Bibliothek.DANumber(Wert_Y_Achse);
// Hinweis: Umrechnung in Pixeleinheit möglich per
// DA_Wert_Y_Achse = DA_Bibliothek.Mul(DA_Wert_Y_Achse,
// DA_Bibliothek.Pixel
// );

// ----- und Lage als animiert erzeugen
var DA_Punkt_Animiert = DA_Bibliothek.Translate2Anim(DA_Wert_X_Achse,DA_Wert_Y_Achse);

// +++++ Rectangle als animiert erzeugen
DA_Rectangle = DA_Rectangle.Transform(DA_Punkt_Animiert);

// +++++ Rectangle mit Eventsteuerung versehen
// ----- Fähigkeit der Eventerzeugung einbinden
DA_Rectangle = DA_Rectangle.PickableOccluded();

// ----- Event MouseOver zum Rectangle erzeugen
var DA_Event_MouseOver_BeiRectangle = DA_Rectangle.PickEvent;

// ----- Event OnClick als Druck der linken Maustaste zum Rectangle hinzufügen
var DA_Event_MouseOver_Und_DruckLinkeMouseTaste_BeiRectangle =
    DA_Bibliothek.AndEvent( DA_Event_Druck_LinkerMaustaste,
        DA_Event_MouseOver_BeiRectangle
    );

// ----- und die Eventbehandlungs-Routine einbinden
var DA_EventHandler_BeiRectangle =
    DA_Event_MouseOver_Und_DruckLinkeMouseTaste_BeiRectangle.ScriptCallback(
        "UrlAusfuehren()",
        "JScript"
    );

// +++++ komplettes DA-Rectangle der Animation erzeugen
DA_Rectangle_Animation = DA_Rectangle.Image;

// ##### DA-Bild erzeugen #####
// +++++ Bild importieren
var DA_Bild = DA_Bibliothek.ImportImage(BildDateiMitPfad);

// +++++ DA-Bild mit Hintergrund versehen
DA_Bild_MitHintergrund = DA_Bibliothek.Overlay(DA_Bild,DA_FuellFarbe_Black);

```



```

// +++++ DA-Bild mit Eventsteuerung versehen
// ----- Fähigkeit der Eventerzeugung einbinden
var DA_Bild_MitEventErzeugung_BeiMouseOver = DA_Bild.PickableOccluded();

// ----- Event MouseOver beim Bild erzeugen
var DA_Event_MouseOver_BeiBild = DA_Bild_MitEventErzeugung_BeiMouseOver.PickEvent;

// ----- Event Kein MouseOver beim Bild erzeugen
var DA_Event_Kein_MouseOver_BeiBild = DA_Bibliothek.NotEvent(DA_Event_MouseOver_BeiBild);

// ##### leeres DA-Bild der Animation bei MouseOver erzeugen #####
var DA_Bild_Animation = DA_Bild_MitEventErzeugung_BeiMouseOver.Image;

// ##### leeres DA-Bild der Animation mit Sound-off verbinden
var Feld = new Array(DA_Bild_Animation, DA_Sound_Off);
var DA_Bild_Animation_Mit_SoundOff = DA_Bibliothek.DATuple(Feld);

// ##### DA-Rectangle, DA-Text und DA-Bild kombinieren
var KomponentenFeld = new Array (
    DA_Rectangle_Animation,
    DA_Text_Animation,
    DA_Bild_Animation
);

var DA_Komposition = DA_Bibliothek.OverlayArray(KomponentenFeld);

// ##### DA-Komposition aus DA-Rectangle, DA-Text und DA-Bild mit Sound verbinden
var Feld = new Array(DA_Komposition, DA_Sound);
DA_Komposition_MitSound = DA_Bibliothek.DATuple(Feld);

// ##### leeres DA-Bild mit Sound-off verbinden #####
Feld = new Array(DA_Leeres_DA_Bild, DA_Sound_Off);
var DA_Bild_Leer_Mit_SoundOff = DA_Bibliothek.DATuple(Feld);
DA_Bild_Leer_Mit_SoundOff = DA_Bibliothek.UninitializedTuple(DA_Bild_Leer_Mit_SoundOff);

// ##### Eventhandler des Rectangle mit Sound off verbinden
//           tue solange
//           Sound off setzen UND leeres Bild überlagern
//           bis Aufruf des Eventhandler erfolgt
//           nach Schleifenende Sound off setzen
DA_EventHandler_BeiRectangle_MitSoundOff = DA_Bibliothek.Until(
    DA_Bild_Leer_Mit_SoundOff, // Zeiger auf Feld
    DA_EventHandler_BeiRectangle, // Zeiger auf Methode
    DA_Bild_Leer_Mit_SoundOff
);

// ##### Steuerung der Animation erzeugen #####
//           verschachtelte Schleifen

// ----- innere Schleife:
//           tue solange
//           Sound erzeugen bei sichtbaren DA-Rectangle, DA-Text und DA-Bild
//           bis kein MouseOver mehr erkannt wird
//           nach Schleifenende: Leeres Bild auf die Komposition legen, also Komposition unsichtbar
//           machen UND Sound abschalten
var DA_Until_Innen = DA_Bibliothek.Until(
    DA_Komposition_MitSound,
    DA_Event_Kein_MouseOver_BeiBild,
    DA_Bild_Leer_Mit_SoundOff
);

// ----- äußere Schleife:
//           tue solange
//           DA-Bild anzeigen und Sound off setzen
//           bis MouseOver über DA-Bild erkannt wird
//           nach Schleifenende: Aufruf der inneren Until-Schleife
DA_Until_InnenUndAussen = DA_Bibliothek.Until(
    DA_Bild_Animation_Mit_SoundOff,
    DA_Event_MouseOver_BeiBild,
    DA_Until_Innen
);

// ##### Steuerung der Animation einbinden #####
DA_Bild_Leer_Mit_SoundOff.Init(DA_Until_InnenUndAussen);

// ##### DA-Init #####
var Index = 0;

```



```

var DA_ElementImDA_Feld = DA_EventHandler_BeiRectangle_MitSoundOff.Nth(Index);
// Index 0 von DA_Bild_Leer_Mit_SoundOff
// DA_EventHandler_BeiRectangle_MitSoundOff ist Zeiger auf Schleife

DAControl.Image = DA_Bibliothek.Overlay(DA_ElementImDA_Feld,DA_Bild_MitHintergrund);

Index = 1;
DA_ElementImDA_Feld = DA_EventHandler_BeiRectangle_MitSoundOff.Nth(Index);
// Index 1 von DA_Bild_Leer_Mit_SoundOff
// DA_EventHandler_BeiRectangle_MitSoundOff ist Zeiger auf Schleife
DAControl.Sound = DA_Bibliothek.Mix(DA_ElementImDA_Feld,DA_Sound_Off);

// nur bei IE 3.x
DAControl.BackgroundImage = DA_FuellFarbe_Blue;

// ##### Start der Animation #####
DAControl.Start();
}
-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
  <OBJECT ID="DAControl"
    CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
    STYLE="position:absolute; left:20; top:75; width:350; height:250;"
  >
  </OBJECT>
  <BR>
  Mit Maus ueber das Bild fahren und klicken!
</FONT>
</BODY>
</HTML>

```

5.2.2.3.9. Zufallswert

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
var KaroGroessenFaktor          = 2;    // >= 1, je größer umso größer der Karodimension
var KaroAbstandFaktor           = 6;    // >= 1, je höher um so mehr Abstand
var KaroAnzahl                  = 10;    // >= 1

var KaroFarbe_RGB_Anteil_Rot     = 15;
var KaroFarbe_RGB_Anteil_Gruen   = 15;
var KaroFarbe_RGB_Anteil_Blau    = 15;

var KaroPositionsWechsel_Traegheit = 2;    // >= 1, je höher um so traeger

var AnimationZentrierungsFaktor  = 35; // >= 1 , je höher, um so zentrierter

function DirectAnimationStart()
{
  function ZufallsWert_Erzeugen()
  {return DA_Bibliothek.DANumber(Math.random());}

  function ZufallsWert_AlsSchnappSchuss_Erzeugen()
  {
    var DA_ZufallWert = DA_Bibliothek.SeededRandom(Math.random());
    return DA_Bibliothek.Always.Snapshot(DA_ZufallWert);    // Snapshot() löst Event aus
  }

  function ZufallsWerteAlsFolgeErzeugen(DauerDerZufallsFolge)
  {
    // Hinweis: Für bessere Zufälligkeit ist Math.random() immer direkt bzw. als
    //           Funktionsaufruf kodieren
    //           und nicht als Variablenwert zwischen zu speichern

    // +++++ DA-Timer laut DauerDerZufallsFolge erzeugen
    var DA_Nicht_Endlos_Timer = DA_Bibliothek.Timer(DauerDerZufallsFolge);

    // +++++ Schleife erzeugen
    //           solange ZufallsWert erzeugen bis Schnappschuss-Event eintritt
    var DA_Schleife = DA_Bibliothek.UntilEx(
      ZufallsWert_Erzeugen(),
      ZufallsWert_AlsSchnappSchuss_Erzeugen()
    );
  }
}

```



```

    );

    // +++++ sich selbständig per Rekursion veränderten DA-Wert erzeugen
    // ----- Objekt erzeugen
    var DA_Number_Rekursion = new ActiveXObject("DirectAnimation.DANumber");

    // ----- Rekursion per Schleife erzeugen
    //         tue solange
    //             Endlosschleife aufrufen
    //         bis DA_Nicht_Endlos_Timer abgelaufen ist
    //         nach Schleifenende aktiviere DA_Number_Rekursion, also Rekursion ausführen

    var DA_Rekursion_Schleife = DA_Bibliothek.Until( DA_Schleife,
                                                    DA_Nicht_Endlos_Timer,
                                                    DA_Number_Rekursion
                                                    );

    // ----- Rekursionsschleife zuweisen per Init
    DA_Number_Rekursion.Init(DA_Rekursion_Schleife);

    // +++++ Zeiger auf DA-Number mit selbständiger Rekursion liefern
    return DA_Number_Rekursion;
}

// +++++ Korrektur von Script-Werten
KaroGroessenFaktor          /= 1000;
var KaroGroessenFaktor_Negativ = -1 * KaroGroessenFaktor;

KaroAbstandFaktor           /= 100;

AnimationZentrierungsFaktor /= -1000;

// ##### DA-Bibliothek #####
var DA_Bibliothek = DAControl.MeterLibrary;

// ##### Zentrum der Animation festlegen #####
var DA_2DPunkt_AnimationZentrum = DA_Bibliothek.Translate2(AnimationZentrierungsFaktor,
                                                            AnimationZentrierungsFaktor
                                                            );

// ##### DA-Bild #####
// +++++ DA-Bild-Gesamt leer erzeugen
var DA_Bild_Gesamt = DA_Bibliothek.EmptyImage;

// +++++ Abstandsfaktor der Karos erzeugen
var DA_KaroAbstandFaktor = DA_Bibliothek.DANumber(KaroAbstandFaktor);

// +++++ Dimension des Karos erzeugen
var DA_2DPunkt1 = DA_Bibliothek.Point2(KaroGroessenFaktor_Negativ,KaroGroessenFaktor_Negativ);
var DA_2DPunkt2 = DA_Bibliothek.Point2(KaroGroessenFaktor,KaroGroessenFaktor);

// +++++ karoweise erzeugen und karoweise dem Gesamtbild hinzufügen
for(var i=0; i < KaroAnzahl; i++)
{
    // - - - DA-Farbe erzeugen aus zufälligen RGB-Anteilen
    var DA_ZufallFarbe_Animiert = DA_Bibliothek.ColorRgbAnim(
        ZufallsWerteAlsFolgeErzeugen(KaroFarbe_RGB_Anteil_Rot),
        ZufallsWerteAlsFolgeErzeugen(KaroFarbe_RGB_Anteil_Gruen),
        ZufallsWerteAlsFolgeErzeugen(KaroFarbe_RGB_Anteil_Blau)
    );

    // - - - DA-Füllfarbe aus der zufälligen DA-Farbe erzeugen
    var DA_ZufallFuellFarbe_Animiert = DA_Bibliothek.SolidColorImage(DA_ZufallFarbe_Animiert);

    // - - - DA-Bild als Karo erzeugen und mit Füllfarbe füllen
    var DA_Bild_Neu = DA_ZufallFuellFarbe_Animiert.Crop(DA_2DPunkt1,DA_2DPunkt2);

    // - - - Zufallsposition des Karo ermitteln
    //         wichtig: Obwohl nachfolgende Berechnungen der Produkte identisch sind,
    //         erfolgen diese mit verschiedenem Wert laut Math.random()
    var DA_Produkt1 = DA_Bibliothek.Mul(
        ZufallsWerteAlsFolgeErzeugen(KaroPositionsWechsel_Traegheit * Math.random()),
        DA_KaroAbstandFaktor
    );

```




```

        var DA_Produkt2 = DA_Bibliothek.Mul(
            ZufallsWerteAlsFolgeErzeugen(KaroPositionsWechsel_Traegheit * Math.random()),
            DA_KaroAbstandFaktor
        );

        var DA_2DPunkt_Zufall = DA_Bibliothek.Translate2Anim(DA_Produkt1, DA_Produkt2);

        // - - - Karo zufällig positionieren aber relativ zum Zentrum der Animation
        var DA_2DKomposition = DA_Bibliothek.Compose2( DA_2DPunkt_Zufall,
            DA_2DPunkt_AnimationZentrum
        );

        // - - - und Karo nach 2D konvertieren
        DA_Bild_Neu = DA_Bild_Neu.Transform(DA_2DKomposition);

        // - - - Karo dem DA-Bild-Gesamt hinzufügen
        DA_Bild_Gesamt = DA_Bibliothek.Overlay(DA_Bild_Gesamt, DA_Bild_Neu);
    }

    // ##### DA-Init #####
    DAControl.Image = DA_Bild_Gesamt;

    // ##### Start der Animation #####
    DAControl.Start();
}

-->
</SCRIPT>
</HEAD>
<BODY onload="DirectAnimationStart();">
    <OBJECT ID="DAControl"
        CLASSID="CLSID:B6FFC24C-7E13-11D0-9B47-00C04FC2F51D"
        STYLE="position:absolute; left:30%; top:100;width:300px;height:300px"
    >
    </OBJECT>
</BODY>
</HTML>

```

5.2.3. JScript Laufzeit-Bibliothek (ActiveXObject) des Internet Explorer

Das Objekt ermöglicht den Zugriff auf die JScript-Laufzeit-Bibliothek als Erweiterung von JScript.

Dieses Objekt ist in der JScript-Maschine aufgrund von Active-X integriert und hat eigentlich nichts mit der DOM-Hierarchie des HTML-Dokumentes zu tun.

Die JScript-Erweiterung berührt auch das DOM, denn alle Bibliotheken-Elemente sind in eine Webseite implementierbar.

Es gibt zwei Hauptkomponenten der Bibliothek:

Dictionary	als Verwaltung von per Schlüssel indizierten Stringdaten per JScript als Addon zum Internet Explorer analog zur Datenbankverwaltung per Textdatei
FileSystemObject	als Verwaltung des Dateisystems im Betriebssystem per JScript als Addon zum Internet Explorer analog zu einer abgespeckten Dateiverwaltung per DOS

Warnung zum FileSystemObject:

Die Firewall müsste Zugriffe beanstanden (Anfrage auf Erlaubnis der Zugriffe auf die Festplatte).

Die Nutzung dieses Objektes berührt unmittelbar die Privatsphäre des Users. Die Nutzung des Objektes kann rechtlich relevant werden, so dass für den User Transparenz bezüglich der Objektverwendung vorliegen muss.

Ein User, der das Active-X-Control mit oder ohne sein Wissen zulässt, geht definitiv das Risiko einer missbräuchlichen Nutzung des Objektes ein, wenn kein Virenschanner die Scripte vor deren Aktivierung prüft und nach User-Entscheidung oder generell blockiert und somit das Privateigentum des Users schützt.

Eine aktive Firewall mit integriertem Virenschanner kann Scriptanweisungen mit Bezug auf das FileSystemObject als virenverseuchtes Script erkennen und die Scriptaktivierung verhindern. Falls der Virenschanner auch E-Mail scannen kann, in denen o.g. Scriptanweisungen vorliegen, könnte eine betroffene Email automatisch bereinigt und damit inhaltlich verändert werden. Der Test o.g. Scriptanweisungen ist also nur bei abgeschalteter Firewall und Virenschanner möglich. Scriptanweisungen mit Bezug auf das FileSystemObject sind absolute Ausnahme im Webdesign, so dass eine Webseite, die unkommentiert das FileSystemObject benutzt, als Virenangriff angesehen werden sollte.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers



Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read"-Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Popufenster geblockt. Sie können den Popupblocker deaktivieren

oder Popsups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter



Popupblocker einschalten
 weitere Informationen
 jedoch keine Möglichkeit wie laut Bedeutung
 Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.
 Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)
 Der Popupblocker des IE bemerkt aber NUR Webseite, die das Popup erzeugt.
 Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.
 Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).
 Der Popupblocker ist nicht als Filter aufgesetzt sondern eingestrickt worden.
 Der Popupblockerfehler verändert die Eventverwaltung:
 Es werden u.a. ignoriert
 onfocus
 onblur
 onfocusin
 onfocusout
 und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

 // nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
 window.focus();
 window.document.focus();
 if(document.body!=null)
 {if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
 }
 // wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
 popupzeiger.show(...);

Nachfolgend eine Testdokument

```

<BODY>
<SCRIPT LANGUAGE="JScript">
// #####
//
//      Quelltext NICHT ändern wegen Fehlermeldung (siehe unten)
//
// #####

// Warnung: window.popup arbeitet im IE 7 fehlerhaft wegen Popupblocker-Fehler:
//      Bedingung:  Scriptfehleranzeige ist erlaubt im IE 7
//                  Popupblocker ist abgeschaltet
//                  Ein Fenster mit Dokument aktiv, dass fortlaufend genau 1 window.popup neu erzeugt (per Rekursion)
//                  Ein paralleles Fenster z.B. Leere Seite (about:blank) oder mit echtem Internetzugriff
//                  das dauerhaft Focus haben muss
//                  Beide Fenster sind Register in einer gemeinsamen IE-Instanz
//      Solange Focus: Irgendwann erscheint eine Meldung des IE 7 per gelber Zeile
//                  im Fenster, das das window.popup erzeugt (Meldung kommt nicht, wenn
//                  Focus auf Fenster, das das window.popup erzeugt)
//
//      'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'
//
//      Bedeutung laut Microsoft-Hilfe:
//      Der Popupblocker hat ein Pop-up-Fenster geblockt. Sie können den Popupblocker deaktivieren
//      oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.
//
//      Realität: linke oder rechte Maus liefert       z.B.       Einstellungen darunter Popupblocker einschalten
//                                                         weitere
Informationen
//
//      jedoch keine Möglichkeit wie laut Bedeutung
//
//      Damit gilt: Der abgeschaltete Popupblocker ist aktiv.
//
//      Blockiertes window.popup:
//
//      Blockierung stoppt Abarbeitung des Scriptes, das das popup rekursiv erzeugt (stopp der
Rekursion)
//      und bewirkt zugleich Meldung eines Scriptfehlers an Quelltext-Stelle zu .show()
//
// #####
//
//      Es wird Zeile 55 in Spalte 2 als Fehlerstelle ausgegeben: Unbekannter Fehler
  
```



```
//
// #####

var XPopupZeiger=window.createPopup();
XPopupZeiger.document.body.innerHTML='##### Test #####';
var XLeft=700;
var XTop=400;
var XTimeoutID1=0;
var XTimeoutID2=0;
var XAnzeigeErlaubt=false;

function Y_PopupShow()
{XLeft-=10;if(XLeft<50){XLeft=700;}
XTop-=10;if(XTop<50){XTop=400;}
XPopupZeiger.show(XLeft,XTop,300,300,document.body);
// Popup wird immer innerhalb Screen-Dimension angezeigt, wenn XLeft und /oder XTop
// ausserhalb der Screen-Dimension
if(XAnzeigeErlaubt){ XTimeoutID1=window.setTimeout('Y_PopupHide();',3000);}
}

function Y_PopupHide()
{XPopupZeiger.hide();
if(XAnzeigeErlaubt){XTimeoutID2=window.setTimeout('Y_PopupShow();',1000);}
}

function YAnzeigeStarten()
{XAnzeigeErlaubt=true;
Y_PopupShow();
}

function YAnzeigeStoppen()
{XAnzeigeErlaubt=false;}
</SCRIPT>

<INPUT type="button" value="Start" onclick="YAnzeigeStarten();">
<BR>
<BR>
<INPUT type="button" value="Stop" onclick="YAnzeigeStoppen();">
</BODY>
```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
zwischen Register in einem IE-Fenster
zwischen Fensters z.B. in Taskleiste

Hinweis:

- .focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
- .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
- funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.



Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:• http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:• <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497} Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp (http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)



verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:•

<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement {05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab



Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: •



<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>

(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

5.2.3.1. Dictionary Objekt

Objekt zur Datenspeicherung in feldähnlicher Form (Dictionary)

Aufbau der Date:

Schlüsselfeld	String
	Index im Dictionary
	eindeutig im gesamten Dictionary:
	Es muss durch den Programmierer die Eindeutigkeit per Methode .Exists() geprüft werden, da Fehlertoleranz im Dictionary-Objekt nicht vorgesehen ist. Bei Zugriff auf einen nicht vorhandenen Schlüssel bzw. bei Einfügung eines bereits im Dictionary vorhandenen Schlüssels erfolgt umgehend die Erzeugung eines Laufzeitfehlers.
Datenfeld	String

Syntax:

```
[ var Zeiger = ] new ActiveXObject("Scripting.Dictionary")
```

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchlüssel, Date);

    // und Meldung ob Date vorhanden ist
    alert(SchlüsselVorhanden(DatenSchlüssel));
}
```

Eigenschaften:

.Count	Anzahl der Elemente im Dictionary
	Integer, ab 1

Methoden:

.Add()	Date zum Dictionary Objekt hinzufügen
--------	---------------------------------------

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchlüssel, Date);

    // und Meldung ob Date vorhanden ist
    alert(SchlüsselVorhanden(DatenSchlüssel));
}
```

.Exists()	prüfen auf Vorhandensein eines Datenschlüssels im Dictionary
-----------	--

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
```



```

{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Meldung ob Date vorhanden ist
    alert(SchluesselVorhanden(DatenSchluessel));
}

```

.Item() Inhalt des Datenfeldes anhand Schlüssel liefern

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Date anzeigen
    alert(DatenSpeicher.Item(DatenSchluessel));
}

```

.Items() Inhalte aller Datenfelder des Dictionary-Objektes als Referenz liefern, die als Zeiger für den Konstruktor new VBArray() (VisualBasic-Anweisung) dient
Datenfelder-Reihenfolge laut Folge im Dictionary

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Daten-Referenz bilden
    var DatenReferenz = DatenSpeicher.Items();

    // und Feld bilden
    // VisualBasic-Feld erzeugen
    var DatenOhneSchluessel_Feld = new VBArray(DatenReferenz);
    // und nach JScript-Feld konvertieren
    DatenOhneSchluessel_Feld = DatenOhneSchluessel_Feld.toArray();

    // und Daten anzeigen
    var DatenOhneSchluessel_FeldLaenge = DatenOhneSchluessel_Feld.length

    if (DatenOhneSchluessel_FeldLaenge > 0)
    {
        for (var i = 0 ; i < DatenOhneSchluessel_FeldLaenge; i++)
        { alert("Date " + i + " = " + DatenOhneSchluessel_Feld [i]); }
    }
    else
    { alert("Dictionary ist leer!"); }
}

```

.Key() Datenschlüssel im Dictionary ändern

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)

```



```

{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // dann Schlüssel ändern
    DatenSpeicher.Key(DatenSchluessel) = "b";

    // und Meldung ob Date vorhanden ist
    alert(SchluesselVorhanden("b"));
}

```

.Keys() Inhalt aller Schlüsselfelder des Dictionary-Objektes als Referenz liefern, die als Zeiger für den Konstruktor new VBAArray () (VisualBasic-Anweisung) dient
Schlüsselfelder-Reihenfolge laut Folge im Dictionary

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Daten-Referenz bilden
    var SchluesselReferenz = DatenSpeicher.Keys();

    // und Feld bilden
    // VisualBasic-Feld erzeugen
    var SchluesselOhneDaten_Feld = new VBAArray(SchluesselReferenz);
    // und nach JScript-Feld konvertieren
    SchluesselOhneDaten_Feld = SchluesselOhneDaten_Feld.toArray();

    // und Daten anzeigen
    var SchluesselOhneDaten_FeldLaenge = SchluesselOhneDaten_Feld.length

    if (SchluesselOhneDaten_FeldLaenge > 0)
    {
        for (var i = 0 ; i < SchluesselOhneDaten_FeldLaenge; i++)
        {alert("Schluessel " + i + " = " + SchluesselOhneDaten_Feld [i]);}
    }
    else
    {alert("Dictionary ist leer!");}
}

```

.Remove() genau eine Date aus dem Dictionary Objekt entfernen (Schlüssel- und Datenfeld entfernen)

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden(DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen

```



```

        DatenSpeicher.add (DatenSchluessel, Date);

        // und Date anzeigen
        alert(DatenSpeicher.Item(DatenSchluessel));

        // und löschen
        DatenSpeicher.remove(DatenSchluessel);

        // und Meldung
        alert(SchluesselVorhanden(DatenSchluessel));
    }

```

.RemoveAll() alle Daten aus dem Dictionary entfernen (alle Schlüssel- und Datenfelder), so dass das Dictionary Objekt leer aber weiterhin instanziiert ist

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden(DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Date anzeigen
    alert(DatenSpeicher.Item(DatenSchluessel));

    // und löschen
    DatenSpeicher.removeAll();

    // und Meldung
    alert(SchluesselVorhanden(DatenSchluessel));
}

```

5.2.3.2. *FileSystemObject Objekt*

Objekt des abgespeckten Dateisystems im Betriebssystem.

Der Zugriff auf das Dateisystem über dieses Objekt hängt teilweise vom Betriebssystem bzw. von Einstellungen zur Ordnen und Dateien ab.

Das FileSystemObject Objekt besitzt weitere Objekte sowie interne Collectionen. Diese müssen per Script durch Ableitung von FileSystemObject instanziiert werden und sind erst danach verfügbar. Abgeleitete Objekte besitzen eigene Eigenschaften und Methoden und bekommen **nichts** vom FileSystemObject vererbt. Damit ist sichergestellt, dass der Zugriff auf das Dateisystem nur im jeweiligen Context des **instanziierten** Objektes stattfindet. Mit anderen Worten: Es existiert kein pauschaler kompletter Zugriff auf das Dateisystem außerhalb der Eigenschaften und Methoden von FileSystemObject, wenn kein weiteres Objekt abgeleitet wurde.. Ein Programmierer, der ableitet, **will** also den Zugriff auf das Dateisystem **bewusst** erweitern.

Der Zugriff auf interne Collectionen ist z.T. nur über das JScript-Objekt Enumerator möglich.

Warnungen:

Die Nutzung dieses Objektes berührt unmittelbar die Privatsphäre des Users. Die Nutzung des Objektes kann rechtlich relevant werden, so dass für den User Transparenz bezüglich der Objektverwendung vorliegen muss.

Es kann sein, dass ein aktiver Virens Scanner, der Scripte in Webseiten untersucht, Virenmeldungen erzeugt.

Ein User, der das Active-X-Control mit oder ohne sein Wissen zulässt, geht definitiv das Risiko einer missbräuchlichen Nutzung des Objektes ein, wenn kein Virens Scanner die Scripte vor deren Aktivierung prüft und nach User-Entscheidung oder generell blockiert und somit das Privateigentum des Users schützt.

Sämtliche Beispiele sind mit Vorsicht zu genießen und möglichst nicht zu testen, es sei denn, der Programmierer weiss genau, was er bewirken will ! In einigen Beispielen wird das Laufwerk C: derart manipuliert, dass bei fehlerhafter Programmierung ein totaler Datenverlust eintreten kann. Das Testsystem sollte also eine Kopie des ursprünglichen Systems sein, das alternativ vorher mit einem Backup-Programm komplett gesichert werden sollte ! Desweiteren müssen eine aktive Firewall und deren Virens Scanner deaktiviert sein.

Ziel der Beispiele ist es nur, die Gefährlichkeit der Verwendung des Objektes FileSystemObject zu zeigen.

Syntax:

```
[ var Zeiger = ] new ActiveXObject("Scripting.FileSystemObject");
```



Beispiel Laufwerke auf dem PC des Users ermitteln (ein harmloses Beispiel):

```
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd(); DateiSystem_Laufwerke.moveNext() )
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        {
            LaufwerkName = "[Drive not ready]";
        }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);
```

Eigenschaften:

keine

Methoden:

Hinweise: Alle Angaben zu einer Datei mit Pfad dürfen nicht mit "\" bzw. "\\" enden, da sonst ein Ordnername erkannt wird.
 Methoden zu Ordnern müssen bei Pfadangaben kein "\" bzw. "\\" am Ende kodiert haben
 Verwendung von Wildcard "*" ist nur z.T. möglich. Bei Ordnern darf "*" nur am Ende einer Pfadanangabe stehen.
 Pfade sind fast immer relativ und absolut kodierbar (Relativ z.B. per "\\.").

.BuildPath() PATH-Variable im Dateisystem erweitern durch Anhängen
 keine Erweiterung des Dateisystems per Ordner (dafür gibt es andere Methoden)

.CopyFile() Datei(en) kopieren
 Wenn Copy abbricht, so bleiben bisher erfolgte Kopieraktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.CopyFile ("c:\\eigene dateien\\*.doc", "c:\\recycled\\")
```

.CopyFolder() Ordner kopieren
 Wenn Copy abbricht, so bleiben bisher erfolgte Kopieraktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.CopyFolder("c:\\*", "d:\\recycled\\")
```

.CreateFolder() Ordner erzeugen, der nicht bereits existieren darf (sonst wird Fehler erzeugt)

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.CreateFolder("c:\\test")
```

.CreateTextFile() Textdatei anlegen und zum Schreiben als Textstream öffnen



Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.CreateTextFile("c:\\test.txt", true);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

.DeleteFile() vorhandene Datei(en) löschen
Wenn Delete abbricht, so bleiben bisher erfolgte Löschaktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.DeleteFile("c:\\*.txt", true);
```

.DeleteFolder() vorhandene Ordner löschen, wobei es egal ist, ob Daten und/oder Unterordner im jeweiligen Ordner liegen oder nicht
Wenn Delete abbricht, so bleiben bisher erfolgte Löschaktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.DeleteFolder("c:\\*", true);
```

.DriveExists() prüfen auf Laufwerk im Dateisystem
Laufwerk muss nicht bereits ein (z.B. Diskettenlaufwerk muss kein Medium enthalten)

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.DriveExists("A"));
```

.FileExists() prüfen auf Datei

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.FileExists("c:\\test.txt"));
```

.FolderExists() prüfen auf Ordner

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.FolderExists("c:\\test\\"));
```

.GetAbsolutePathName() absoluten Pfad ab Root mit Laufwerk ermitteln
Angabe von Wildcard "*" möglich wird **leider nicht** aufgelöst
Angabe von "\\." möglich falls Anzahl von "\\." eine Logik hinter die Root erzeugen würde, so wird kein Fehler erzeugt, sondern korrekt ab Root aufgelöst
Bsp.: c:\\test\\ existiert
Auflösung von c:\\..\\..\\test bringt keinen Fehler

Beispiel 1:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetAbsolutePathName("..\\test\\"));
```

Beispiel 2: Es gilt: aktuelle Position im Dateisystem ist c:\\test\\texte
Ordner texte hat folgenden Unterordner jahr2002
Ordner jahr2002 hat folgenden Unterordner mai2002

<u>Kette1</u>	<u>Kette2</u>	<u>Bedeutung</u>
"c:"	"c:\\test\\texte"	aktuelle Position
"c:.."	"c:\\test"	Ordner oberhalb der aktuellen Position
"c:\\"	"c:\\"	Root und nicht aktuelle Position
"c:.*\\mai2002"	"c:\\test\\texte*.\\mai2002"	Pfad von mai2002
"jahr2002"	"c:\\test\\texte\\jahr2002"	keine Auflösung von "*" Pfad von jahr2002
"c:\\..\\..\\test"	"c:\\test"	kein Fehler

.GetBaseName() absoluten Pfad einer Datei im Pfad liefern
Datei muss nicht existieren

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetBaseName("..\\test\\text.txt"));
```

.GetDrive() Objekt FileSystemObject.Drive anhand des Laufwerksbuchstaben eines existierenden Laufwerkes erzeugen (auch bei Netzlaufwerk) sonst wird Fehler erzeugt

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk = DateiSystem.GetDrive("C");
var Laufwerk_VolumeName = Laufwerk.VolumeName;
```



```
var Laufwerk_TotalePlatz = Laufwerk.TotalSize;
alert(Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalePlatz);
```

.GetDriveName() Laufwerksbuchstabe aus Pfad ermitteln in der Form "x:" mit x für Laufwerk
 Pfad muss nicht existieren
 muss Laufwerksbuchstaben enthalten
 wird nicht auf Syntax geprüft

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
alert(Dateisystem.GetDriveName("c:\\.\\.*\\test\\t***"));
```

.GetExtensionName() Suffix einer Datei liefern ohne Trenner "."
 Datei muss nicht existieren

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
alert(Dateisystem.GetExtensionName("..\\test\\text.txt"));
```

.GetFile() Objekt FileSystemObject.File anhand Dateinamen einer existierenden Datei erzeugen
 (sonst wird Fehler erzeugt)

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = Dateisystem.GetFile("..\\test\\text.txt");
alert(Datei);
```

.GetFileName() Name einer Datei mit Suffix und mit Trenner "."
 Datei muss nicht existieren

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
alert(Dateisystem.GetFileName("..\\test\\text.txt"));
```

.GetFolder() Objekt FileSystemObject.Folder anhand Ordernamen eines existierenden Ordners erzeugen
 sonst wird Fehler erzeugt

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = Dateisystem.GetFolder("..\\test\\");
alert(Ordner);
```

.GetParentFolderName() Elternordner einer Datei liefern
 Datei muss nicht existieren

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
alert(Dateisystem.GetParentFolderName("..\\test\\text.txt"));
```

.GetSpecialFolder() Objekt FileSystemObject.Folder von Ordnern des Betriebssystems erzeugen

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = Dateisystem.GetSpecialFolder(0);
var DateiOffen = Ordner.CreateTextFile("test.txt");
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

.GetTempName() Bezeichner anhand Zufallszahl erzeugen
 Bezeichner verwendbar als Datei- oder Ordner-Bezeichner

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = Dateisystem.GetSpecialFolder(2);
var Dateiname = Dateisystem.GetTempName();
var DateiOffen = Ordner.CreateTextFile(Dateiname);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

.MoveFile() Datei(en) verschieben
 Wenn Move abbricht, so bleiben bisher erfolgte Verschiebungen leider erhalten

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
Dateisystem.MoveFile("c:\\eigene dateien\\*.doc", "c:\\recycled\\")
```

.MoveFolder() Ordner verschieben
 Wenn Move abbricht, so bleiben bisher erfolgte Verschiebungen leider erhalten

Beispiel:

```
var Dateisystem = new ActiveXObject("Scripting.FileSystemObject");
Dateisystem.MoveFolder("c:\\eigene dateien\\", "c:\\recycled\\")
```

.OpenTextFile() Datei als Text öffnen zum Lesen, Schreiben oder Append (Schreiben durch Anhängen)



Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0)
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

5.2.3.2.1. FileSystemObject.Drive Objekt

Objekt für Zugriff auf Laufwerke (lokale oder im Netz)

Erzeugung:

```
[ var Zeiger = ] new ActiveXObject("Scripting.FileSystemObject")
```

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_FreierPlatz = Laufwerk.FreeSpace;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_FreierPlatz);
```

Eigenschaften:

.AvailableSpace

Freien Speicher in Bytes auf Laufwerk ermitteln

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_FreierPlatz = Laufwerk.AvailableSpace;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_FreierPlatz);
```

.DriveLetter

Laufwerksbuchstaben liefern (ohne : und \\ etc.)

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Buchstabe = Laufwerk.DriveLetter;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Buchstabe);
```

.DriveType

Laufwerkstyp liefern

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Typ = Laufwerk.DriveType;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Typ);
```

.FileSystem

Typ des Dateisystems auf dem Laufwerk liefern

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Dateisystem = Laufwerk.FileSystem;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Dateisystem);
```

.FreeSpace

Freien Speicher in Bytes auf Laufwerk ermitteln

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_FreierPlatz = Laufwerk.FreeSpace;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_FreierPlatz);
```

.IsReady

Bereitschaft des Laufwerkes für Zugriffe z.B. ob Medium im Laufwerk liegt

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Bereitschaft = Laufwerk.IsReady;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Bereitschaft);
```

.Path

Pfad eines Laufwerkes ermitteln in voller Länge (nicht 8.3 Pfad)



Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Pfad = Laufwerk.Path;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Pfad);
```

.RootFolder Root des Laufwerkes liefern, also mit "\"

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Root = Laufwerk.RootFolder;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Root);
```

.SerialNumber Seriennummer des Laufwerkes liefern

Achtung: Die Seriennummer lässt eine eindeutige Identifizierung der Festplattenhardware zu vor allem dann, wenn zusätzlich Userdaten zum PC bekannt sind.

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_SerienNummer = Laufwerk.SerialNumber;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " +
      + Laufwerk_SerienNummer);
```

.ShareName öffentlicher Netzwerkname des Laufwerkes liefern

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_ShareName = Laufwerk.ShareName;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_ShareName);
```

.TotalSize Gesamten Speicher in Bytes auf Laufwerk ermitteln

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_TotalerPlatz = Laufwerk.TotalSize;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalerPlatz);
```

.VolumeName Volumenbezeichner des Laufwerkes

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_TotalerPlatz = Laufwerk.TotalSize;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalerPlatz);
```

Methoden:

keine

5.2.3.2.2. FileSystemObject.Drives Collection

interne Collection aller Laufwerke, egal ob sie bereit sind oder nicht (z.B. Diskettenlaufwerk muss kein Medium besitzen)
alle Elemente in der Collection sind nur lesbar

Erzeugung:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = DateiSystem.Drives; // interne Collection
var DateiSystem_DrivesCollection = new Enumerator(DateiSystem_Laufwerke);
// durch Programmierer handhabbare Collection
```

Zugriff:

z.T. nur in Verbindung mit JScript-Objekt Enumerator

Beispiel Laufwerke auf dem PC des Users ermitteln (ein harmloses Beispiel):

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
```



```

var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd();DateiSystem_Laufwerke.moveNext()
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]";}
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

Eigenschaften:

.Count Anzahl der Elemente in der Collection FileSystemObject.Drives
Integer, ab 1

Methoden:

.Item() Eintrag in der Collection FileSystemObject.Drives liefern
nur in Verbindung mit JScript-Objekt Enumerator

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem                = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke     = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd();DateiSystem_Laufwerke.moveNext()
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

```



```

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

5.2.3.2.3. FileSystemObject.File Objekt

Objekt für Zugriff auf eine **vorhandene** Datei (sonst Fehler erzeugt)

Erzeugung:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(Kette);

```

Kette String
Dateiname mit Pfad z.B. "c:\\test.txt"

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateCreated);

```

Eigenschaften:

.Attributes Attribute der Datei

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
var Datei_Attribute = Datei.attributes;

```

```

// auf gesetztes Archivattribut prüfen
if (Datei_Attribute && 32)
{
    // löschen
    Datei_Attribute -= 32;
}
else
{
    // setzen
    Datei_Attribute += 32;
}

```

.DateCreated Datum und Zeit der Dateierstellung liefern

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateCreated);

```

.DateLastAccessed Datum und Zeit des letzten Dateizugriffes liefern

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateLastAccessed);

```

.DateLastModified Datum und Zeit der letzten Dateiänderung liefern

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateLastModified);

```

.Drive Buchstaben des Laufwerkes liefern, auf dem die Datei liegt

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);

```



```
alert(Datei.Drive);
```

.Name Dateibezeichner in voller Länge (nicht 8.3 Bezeichner)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.Name);
```

.ParentFolder Bezeichner des Ordners liefern, in dem die Datei liegt

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.ParentFolder);
```

.Path Pfad der Datei liefern in voller Länge (nicht 8.3 Pfad)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.Path);
```

.ShortName Dateibezeichner als 8.3 Bezeichner liefern

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.ShortName);
```

.ShortPath Pfad der Datei als 8.3 Pfad liefern

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.ShortPath);
```

.Size Dateigröße in Bytes liefern

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.Size);
```

.Type Beschreibung zum Suffix des Dateibezeichners liefern laut Registry
z.B. Suffix ist .TXT Beschreibung ist "Text Document"

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
alert(Datei.Type);
```

Methoden:

.Copy() vorhandene Datei kopieren (sonst Fehler erzeugt)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
Datei.Copy("c:\\windows\\desktop\\test2.txt");
```

.Delete() vorhandene Datei löschen (sonst Fehler erzeugt)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
Datei.Delete();
```

.Move() vorhandene Datei verschieben (sonst Fehler erzeugt)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var Dateisystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = Dateisystem.GetFile(DateinameMitPfad);
Datei.Move("c:\\windows\\desktop\\");
```

.OpenAsTextStream() vorhandene Datei als Text öffnen zum Lesen und Schreiben oder Append, wenn Dateiattribut es



zulassen (sonst Fehler erzeugt)

Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```
var DateiNameMitPfad      = "c:\\test.txt";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Datei                 = DateiSystem.GetFile(DateiNameMitPfad);
var DateiOffen            = Datei.OpenAsTextStream(2,0);
DateiOffen.Write( "neuer Text" );
DateiOffen.Close();
DateiOffen                = Datei.OpenAsTextStream(1,2);
var Kette                 = DateiOffen.ReadLine( );
DateiOffen.Close();
alert(Kette);
```

5.2.3.2.4. FileSystemObject.Folder Objekt

Objekt für Zugriff auf eine **vorhandenen** Ordner (sonst Fehler erzeugt)

Erzeugung:

```
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(Kette);
```

Kette String
 Ordner mit Pfad z.B. "c:\\test\\"

```
var SpezialOrdner         = DateiSystem.GetSpecialFolder(Wert)
```

Wert Integer
 0 Windows-Ordner
 1 System-Ordner
 2 Temp-Ordner

Eigenschaften:

.Attributes Attribute des Ordner

Beispiel:

```
var OrdnerName            = "c:\\windows\\desktop\\";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(OrdnerName);
var Ordner_Attribute      = Ordner.attributes;
```

```
// auf gesetztes Archivattribut prüfen
if (Ordner_Attribute && 32)
{
    // löschen
    Ordner_Attribute -= 32;
}
else
{
    // setzen
    Ordner_Attribute += 32;
}
```

.DateCreated Datum und Zeit der Ordnererstellung liefern

Beispiel:

```
var OrdnerName            = "c:\\windows\\desktop\\";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateCreated);
```

.DateLastAccessed Datum und Zeit des letzten Ordnerzugriffes liefern

Beispiel:

```
var OrdnerName            = "c:\\windows\\desktop\\";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateLastAccessed);
```

.DateLastModified Datum und Zeit der letzten Ordneränderung liefern

Beispiel:

```
var OrdnerName            = "c:\\windows\\desktop\\";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateLastModified);
```

.Drive Buchstaben des Laufwerkes liefern, auf dem der Ordner liegt

Beispiel:

```
var OrdnerName            = "c:\\windows\\desktop\\";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = DateiSystem.GetFolder(OrdnerName);
```



```
alert(Ordner.Drive);
```

.Files Zeiger auf die interne Collection FileSystemObject.Folder.Files
Collection kann nur z.T. über das JScript-Objekt Enumerator angesprochen werden

.IsRootFolder prüfen auf Root

Beispiel:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
var Zahler               = 0;
if (!Ordner.IsRootFolder)
{
    do
    {
        Ordner = Ordner.ParentFolder;
        Zahler++;
    }
    while (!Ordner.IsRootFolder);
}

alert("Ordner liegt " + Zahler + " Ebenen unter der Root");
```

.Name Ordnerbezeichner in voller Länge (nicht 8.3 Bezeichner)

Beispiel:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Name);
```

.ParentFolder Bezeichner des Eltern-Ordners liefern, in dem der Ordner liegt

Beispiel 1:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ParentFolder);
```

Beispiel 2:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
var Zahler               = 0;
if (!Ordner.IsRootFolder)
{
    do
    {
        Ordner = Ordner.ParentFolder;
        Zahler++;
    }
    while (!Ordner.IsRootFolder);
}

alert("Ordner liegt " + Zahler + " Ebenen unter der Root");
```

.Path Pfad des Ordners liefern in voller Länge (nicht 8.3 Pfad)

Beispiel:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Path);
```

.ShortName Ordnerbezeichner als 8.3 Bezeichner liefern

Beispiel:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ShortName);
```

.ShortPath Pfad des Ordners als 8.3 Pfad liefern

Beispiel:

```
var OrdnerName          = "c:\\windows\\desktop\\";
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var Ordner               = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ShortPath);
```



.Size Ordnergrösse in Bytes liefern

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Size);
```

.SubFolders Zeiger auf die interne Collection FileSystemObject.Folder.Folders
Collection kann nur z.T. über das JScript-Objekt Enumerator angesprochen werden

.Type Beschreibung zum Suffix des Ordnerbezeichners liefern laut Registry

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Type);
```

Methoden:

.Copy() vorhandenen Ordner kopieren (sonst Fehler erzeugt)

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
Ordner.Copy("d:\\recycled\\");
```

.Delete() vorhandenen Ordner löschen (sonst Fehler erzeugt)

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
Ordner.Delete();
```

.Move() vorhandenen Ordner verschieben (sonst Fehler erzeugt)

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
Ordner.Move("d:\\recycled\\");
```

5.2.3.2.4.1. FileSystemObject.Folder.Files Collection

interne Collection aller Dateien innerhalb eines Ordners (ohne Dateien in Unterordnern)

alle Elemente in der Collection sind nur lesbar

Erzeugung:

```
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Ordner = DateiSystem.GetFolder(Kette);
var DateiSystem_Dateien = DateiSystem_Ordner.Files; // interne Collection
var DateiSystem_FilesCollection = new Enumerator(DateiSystem_Dateien);
// vom Programmierer verwaltbare Collection

Kette String
Pfad und Name des Ordners
```

Zugriff:

z.T. nur in Verbindung mit JScript-Objekt Enumerator

Beispiel:

```
var Ordner          = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Ordner = DateiSystem.GetFolder(Ordner);
var DateiSystem_Dateien = DateiSystem_Ordner.Files;
var DateiSystem_FilesCollection = new Enumerator(DateiSystem_Dateien);
var Kette           = "";
for (; !DateiSystem_FilesCollection.atEnd(); DateiSystem_FilesCollection.moveNext())
{ Kette = Kette + DateiSystem_FilesCollection.item() + "\n"; }
alert(Kette);
```

Eigenschaften:

.Count Anzahl der Elemente in der Collection FileSystemObject.Folder.Files
Integer, ab 1

Methoden:

.Item() Eintrag in der Collection FileSystemObject.Files liefern
nur in Verbindung mit JScript-Objekt Enumerator

Beispiel:

```
var Ordner          = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Ordner = DateiSystem.GetFolder(Ordner);
var DateiSystem_Dateien = DateiSystem_Ordner.Files;
var DateiSystem_FilesCollection = new Enumerator(DateiSystem_Dateien);
```



```

var Kette = "";
for (; ! DateiSystem_FilesCollection.atEnd(); DateiSystem_FilesCollection.moveNext())
{ Kette = Kette + DateiSystem_FilesCollection.item() + "\n"; }
alert(Kette);

```

5.2.3.2.4.2. **FileSystemObject.Folder.Folders Collection**

interne Collection aller Ordner innerhalb eines **Ordners**, wobei **nur die Ordner eine Hierarchiestufe tiefer gemeint sind** !
 alle Elemente in der Collection sind nur lesbar

Erzeugung:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Ordner = DateiSystem.GetFolder(Kette);
var DateiSystem_FoldersCollection = new Enumerator(DateiSystem_Ordner.SubFolders);
// vom Programmierer verwaltbare Collection

```

Kette String
 Pfad und Name des Ordners

Zugriff:

z.T. nur in Verbindung mit JScript-Objekt Enumerator

Beispiel:

```

var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
var DateiSystem_FoldersCollection = new Enumerator(Ordner.SubFolders);
var Kette = "";
for (; ! DateiSystem_FoldersCollection.atEnd(); DateiSystem_FoldersCollection.moveNext())
{ Kette = Kette + DateiSystem_FoldersCollection.item() + "\n"; }
alert(Kette);

```

Eigenschaften:

.Count Anzahl der Elemente in der Collection FileSystemObject.Folder.Folders
 Integer, ab 1

Methoden:

.Item() Eintrag in der Collection Collection FileSystemObject.Folder.Folders liefern
 nur in Verbindung mit JScript-Objekt Enumerator

Beispiel:

```

var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
var DateiSystem_FoldersCollection = new Enumerator(Ordner.SubFolders);
var Kette = "";
for (; ! DateiSystem_FoldersCollection.atEnd(); DateiSystem_FoldersCollection.moveNext())
{ Kette = Kette + DateiSystem_FoldersCollection.item() + "\n"; }
alert(Kette);

```

.Add() Ordner der Collection Collection FileSystemObject.Folder.Folders hinzufügen
 Ordner darf in der Collection nicht bereits existieren (sonst Fehler erzeugt)

Beispiel:

```

var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
var DateiSystem_FoldersCollection = Ordner.SubFolders;
var NeuerOrdner = DateiSystem_FoldersCollection.Add("Neuer Ordner");

```

5.2.3.2.5. **FileSystemObject.TextStream Objekt**

Objekt einer sequentiellen und offenen Text-Datei auf Basis eines Textstreams (Datenfluss aus Textzeichen), der

intern zeilenweise (durch Programmierer nicht veränderbar)
 extern zeilen- und/oder zeilenweise (durch Programmierer einstellbar)
 verwaltet wird.

Datei ist lesbar **oder** schreibbar (beides gleichzeitig geht leider **nicht**):

satzweise: zeilenweise
 es muss Zeichen newline
 per .WriteLine() bzw. .WriteBlankLines() geschrieben worden sein
 gelesen worden sein
 zeilenweise: wenn newline-Zeichen per .WriteBlankLines() geschrieben wurde, so wird Satzendemarke
 erzeugt
 zeichen- und satzweise: ist möglich, denn ein Zeilenende wird nur erkannt, wenn newline-Zeichen gelesen wurde
 Achtung: zeilenweises Lesen per .ReadLine() erwartet entweder newline-Zeichen oder
 Dateieinde (Textstream-Ende)
 als komplette Datei nur Lesen per .ReadAll()
 (alle Daten auf einen Schlag lesen, egal ob die Datei zeichen- oder zeilenweise
 erstellt wurde)

interner Satzzeiger verfügbar: bei zeichweisem Lesen bzw. Schreiben ist es ein **Zeichenzeiger**
 bei zeilenweisem Lesen bzw. Schreiben ist es ein **Zeilenzeiger**
 wird nicht verwendet, wenn Datei auf einen Schlag komplett gelesen wird per .ReadAll()



numerischer Zeichenzeiger verfügbar (nur lesen): nur bei zeichenweiser Dateiverarbeitung (in der Datei darf kein newline-Zeichen auftauchen)

Die Daten der Textdatei werden im Hauptspeicher gepuffert (Größe des verfügbaren RAM bei Größe der Textdatei beachten).

Hinweis: Neu anlegen **und öffnen** einer Text-Datei per Methode .CreateTextFile() des Objektes FileSystemObject

Erzeugung:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile(OpenTextFile(Kette [, Wert 1[, Wert2 [,Wert3]]])
```

Kette	String	
	Pfad (relativ oder absolut) mit	
	Dateiname ohne Wildcards	
Wert1	Integer	
	1	zum Lesen
	2	zum Schreiben
	8	für Append
Wert2	Boolean	
	True	Datei wird erzeugt, wenn nicht vorhanden
	False	Datei wird nicht erzeugt, muss also vorhanden sein
		(sonst wird Fehler erzeugt)
		Standard
Wert2	Integer	
	0	als ASCII öffnen
	1	als Unicode öffnen
	2	Codeart beim Öffnen laut Systemeinstellung
		Standard
Zeiger	auf offene Datei	

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

Eigenschaften:

.AtEndOfLine Satzzeiger bezüglich Zeilenende (End Of Line (EOL)) per newline-Zeichen
wird mit jedem Lesen oder Schreiben verändert
Hinweis: newline-Zeichen per .WriteLine oder .WriteBlankLines() schreibbar
immer bei satzweisem Schreiben/Lesen
möglich beim zeichenweisem Schreiben/Lesen

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
var Kette = "";
while (!DateiOffen.AtEndOfLine)
{ Kette = Kette + DateiOffen.ReadLine() + "\n"; }
DateiOffen.Close();
alert(Kette);
```

.AtEndOfStream Satzzeiger bezüglich Dateiende, also Textstream-Ende (End Of Stream (EOS))
wird mit jedem Lesen oder Schreiben verändert

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
var Kette = "";
while (!DateiOffen.AtEndOfStream)
{ Kette = Kette + DateiOffen.ReadLine() + "\n"; }
DateiOffen.Close();
alert(Kette);
```

.Column aktuelle Spalte des Zeichens im Textstream, also Nummer des Zeichens im Stream
nur bei **zeichenweiser** Dateiverarbeitung verwendbar, das jedes gelesene newline-Zeichen
den Wert von .Column auf 1 setzt

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
```



```
DateiOffen.ReadLine();
alert(DateiOffen.Column);
DateiOffen.Close();
```

.Line Anzahl der Zeilen im Textstream
sinnvoll nur verwendbar, wenn mindestens 1 newline-Zeichen in der Datei auftaucht

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.ReadAll();
alert(DateiOffen.Line);
DateiOffen.Close();
```

Methoden:

.Close() offene Datei schliessen
nach Schliessen der Datei sind keine Methoden mehr verwendbar, ohne die Datei vorher neu zu öffnen

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

.Read() in der offenen Datei die nächsten Zeichen lesen (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.Read(6));
DateiOffen.Close();
```

.ReadAll() offene Datei komplett auslesen (auf einen Schlag)
nur direkt nach Dateiöffnen möglich

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.ReadAll());
DateiOffen.Close();
```

.ReadLine() in der offenen Datei die nächste Zeile lesen (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.ReadLine());
DateiOffen.Close();
```

.Skip() in der offenen Datei die nächsten Zeichen überlesen (egal ob newline dabei ist oder nicht)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.Skip(1);
alert(DateiOffen.Read(1));
DateiOffen.Close();
```

.SkipLine() in der offenen Datei die nächsten Zeilen überlesen
verändert den Satzzeiger
sinnvoll nur verwendbar, wenn mindestens 1 newline-Zeichen in der Datei auftaucht

Beispiel:



```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("Test");
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.SkipLine(1);
alert(DateiOffen.ReadLine());
DateiOffen.Close();

```

`.Write()` in die offenen Datei zeichenweise schreiben (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();

```

`.WriteBlankLines()` in die offenen Datei newline-Zeichen schreiben (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.WriteBlankLines(1);
DateiOffen.Close();

```

`.WriteLine()` in die offenen Datei genau 1 Zeile schreiben (ab Position laut Satzzeiger)
wobei am Ende automatisch genau ein newline-Zeichen geschrieben wird
verändert den Satzzeiger

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("123456789");
DateiOffen.Close();

```

5.2.3.2.6. FileSystemObject und Windows Script Host (WSH)

Das File-System-Objekt kann teilweise nur in Verbindung mit dem Windows Script Host verwendet werden.

Dadurch werden z.B. windows-spezifische Ordner wie „Recent“, also spezielle Ordner (special folder), zugänglich.

JScript und WSH arbeiten bestens zusammen.

5.2.3.2.6.1. Beispiel: Zugriff auf Recent-Ordner

Beispiel für die Bereinigung des Recent-Ordners zum aktuellen Benutzer: Entfernen von ungültigen Verknüpfungen im Ordner.

Es soll der Ordner der letzten Dateizugriffe verarbeitet werden,
dessen Name "Recent" ist
der physisch im aktuellen Benutzerprofil liegt z.B. c:\Dokumente und Einstellungen\user_name\Recent
der von Windows verwaltet wird und daher ein Spezial-Ordner (special folder) ist
der Verknüpfungen (Ink-Dateien) auf diejenigen Dateien/Ordner enthält, auf die der Benutzer zuletzt zugegriffen hat

```

// ##### Variablen

// Name des Recent-Ordners auf Festplatte
var SpecialFolderName = 'Recent';

// Suffix der Verknüpfungen aus dem Ordner Recent
// Recent-Ordner enthält nur Verknüpfungen (Shortcuts), als Ink-Dateien
var VerknuepfungDateiSuffix = 'lnk'; // nur Kleinbuchstaben !

// WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_DateiVerknuepfung;
var WinScriptHost_DateiVerknuepfung_ZielPfad;

// JScript-Objekte zum Dateisystem
var JScript_DateiSystem;

var SpecialFolder;
var SpecialFolder_Pfad;
var SpecialFolder_Inhalt;
var SpecialFolder_DateienSammlung;
var SpecialFolder_DateiMitPfad;
var SpecialFolder_DateiMitPfad_Suffix;

```



```

// Zähler
var AnzahlVerknuepfungen_Gefunden = 0;
var AnzahlVerknuepfungen_Geloscht = 0;
var AnzahlVerknuepfungen_ZugriffGesperrt = 0;

// ##### Funktion

function VerknuepfungPruefenUndBereinigen(DateiMitPfad)
// Eine Verknüpfung kann auf eine Datei oder einen Ordner verweisen (Ziel der Verknüpfung).

// Es wird die Verknüpfung laut DateiMitPfad überprüft, ob sie noch gültig ist.
// Gültig: Verknüpfung verweist auf eine real-existierende Datei bzw. Ordner.

// Es kann sein, dass auf Verknüpfungen im Ordner Recent wegen NTFS-Zugriffsbeschränkungen
// (zum aktuellen Benutzer) nicht zugegriffen werden kann.

// Fehlschlagende Zugriffe sollten abgefangen werden: Dazu muss der Zugriff
// in die JScript-eigene Funktion try-catch eingebettet werden, um
// Ausnahmefehler während der Scriptlaufzeit auswerten zu können
// (Error-Exceptions).
{
// +++++ Anzahl der gefundenen Dateien (Verknüpfungen) erhöhen
AnzahlVerknuepfungen_Gefunden++;

// +++++ Annahme: Auf die Verknüpfungsdatei kann zugegriffen werden
var ZugriffGesperrt=false;

// +++++ Aus der Verknüpfung per WinScript einen Zeiger bilden, anhand dem im Script geprüft werden kann.
try // versuche Verknüpfung per WinScript zu erzeugen
// wenn dabei ein Laufzeit-Fehler auftritt, wird catch-Zweig aktiviert
{
// Verknüpfung als Zeiger erzeugen
WinScriptHost_DateiVerknuepfung = WinScriptHost_LaufzeitUmgebung.CreateShortcut(DateiMitPfad);

// +++++ aus erzeugter Verknüpfung den Pfad, auf den die Verknüpfung weist, holen
WinScriptHost_DateiVerknuepfung_ZielPfad = WinScriptHost_DateiVerknuepfung.TargetPath

// +++++ überprüfen, ob Zielpfad auf ein nicht-existentes Element verweist (Datei oder Ordner)

// ---- auf nicht-existente Datei prüfen
if (!JScript_DateiSystem.FileExists(WinScriptHost_DateiVerknuepfung_ZielPfad))
{
// Datei nicht existent, aber es könnte ja noch auf einen Ordner verwiesen werden

// ---- auf nicht-existent Ordner prüfen
if (!JScript_DateiSystem.FolderExists(WinScriptHost_DateiVerknuepfung_ZielPfad))
{
// weder Datei noch Ordner real vorhanden, also Verknüpfung
// ist ungültig
// wird entfernt aus dem Ordner Recent

try // versuche ungültige Verknüpfung zu löschen
// wenn dabei ein Laufzeit-Fehler auftritt, wird catch-Zweig aktiviert
{
JScript_DateiSystem.DeleteFile(DateiMitPfad);

// Anzahl der Verknüpfungen, die ungültig sind und gelöscht wurden, erhöhen
AnzahlVerknuepfungen_Geloscht++;
}
catch(LoeschenFehlgeschlagen) // Reaktion bei einfangenem (ge-catchten) Fehler
{ZugriffGesperrt=true;}
}
}
}
catch(VerknuepfungFehlgeschlagen) // Reaktion bei einfangenem (ge-catchten) Fehler
{ZugriffGesperrt=true;}

if (ZugriffGesperrt)
{
// Anzahl der Verknüpfungen, auf die nicht zugegriffen werden kann, erhöhen
AnzahlVerknuepfungen_ZugriffGesperrt++;
}
}
}

```



```
// ##### Prüfen und bereinigen

// ***** Erzeugung der WinScriptHost-LaufzeitUmgebung für Zugriff auf JScript_DateiSystem
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// ***** Erzeugung des DateiSystem als ActiveX per JScript
JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// ***** Recent-Ordner als Objekt per WinScript erzeugt
SpecialFolder = WinScriptHost_LaufzeitUmgebung.SpecialFolders(SpecialFolderName);

// ***** Pfad des Recent-Ordners holen
SpecialFolder_Pfad = JScript_DateiSystem.GetFolder(SpecialFolder)

// ***** Inhalt des Recent-Ordners als Objekt
SpecialFolder_Inhalt = SpecialFolder_Pfad.Files;

// ***** Inhalt des Recent Ordners als Aufzählungsobjekt (Enumerator) mit internem Index
//      interner Index steht auf 1. Element der Aufzählung
SpecialFolder_DateienSammlung = new Enumerator(SpecialFolder_Inhalt)

// ***** Recent-Ordner abklappen
while (!SpecialFolder_DateienSammlung.atEnd()) // solange kein Ende erreicht
{
    // ++++++ aktuelles Element laut internen Index, also Verknüpfungs-Datei, einstellen
    //      und internen Index danach um 1 erhöhen
    SpecialFolder_DateienSammlung.moveNext();

    // ++++++ zum Element den Pfad holen
    var SpecialFolder_DateiMitPfad = SpecialFolder_DateienSammlung.item();

    //      und den Dateisuffix in Kleibuchstaben holen
    SpecialFolder_DateiMitPfad_Suffix = JScript_DateiSystem.GetExtensionName(SpecialFolder_DateiMitPfad);
    SpecialFolder_DateiMitPfad_Suffix = SpecialFolder_DateiMitPfad_Suffix.toLowerCase();

    // ++++++ Element auf lnk-Datei prüfen
    if (SpecialFolder_DateiMitPfad_Suffix == VerknuepfungDateiSuffix)
    { VerknuepfungPruefenUndBereinigen(SpecialFolder_DateiMitPfad); }
}

alert('Im Ordner ' + SpecialFolderName + ' wurden ' + AnzahlVerknuepfungen_Gefunden + ' Verknüpfungen gefunden !');
alert('Im Ordner ' + SpecialFolderName + ' wurden ' + AnzahlVerknuepfungen_Geloscht + ' ungültige Verknüpfungen gelöscht !');
alert('Im Ordner ' + SpecialFolderName + ' sind ' + AnzahlVerknuepfungen_ZugriffGesperrt + ' Verknüpfungen im Zugriff gesperrt !');
```

5.2.3.2.6.2. Beispiel: Zugriff auf Registry

Nachfolgender Quellcode muss in einer eigenständigen JS-Datei untergebracht sein.

Grund: Das Script muss in der Registry registriert werden, damit es per Kontextmenü aktivierbar ist.

Mit der Script-Registrierung wird in das Kontextmenü der Fenster des Windows Explorers einen Eintrag eingefügt, mit dem ein neuer Ordner erzeugt UND angezeigt werden kann.

Standardname des erzeugten Ordners liegt in der Variablen StandardOrdnerName.

Der Kontextmenü-Eintrag-Text liegt in der Variablen KontextMenuEintragText.

Die Erzeugung des neuen Ordners erfolgt durch Aufruf dieses Scriptes, das dazu BEREITS in der Registry registriert sein muss.

Mit der Script-De-Registrierung wird zugleich der Kontextmenü-Eintrag entfernt.

Scriptaufruf: script_js_date [ordner_name]

ordner_name: optional
 Name des Ordners in dem ein Unterordner erzeugt werden soll
 wenn nicht kodiert, so gilt:
 Ist das Script noch nicht in der Registry registriert, so wird es registriert.
 Ist ein Teil der Registry-Schlüssel zum Script nicht vorhanden, so wird das Script neu registriert.
 Ist das Script bereits in der Registry registriert, so wird es de-registriert.

// ##### Variablen

// +++++ Variablen, die frei wählbar sind
 var StandardOrdnerName = "Neuer Ordner";

(TWS) Microsoft JScript für den Hobby-Programmierer




```

var KontextMenuEintragText = "Neuen Unterordner erzeugen";
var Script_RegistryKennung = "TestScript";

// +++++ interne Schlüssel des Script zum Registrieren und De-Registrieren (nicht verändern)
// Lage der Schlüssel in der Registry:
// Directory und Drive für Dateisystem-Verwaltung durch Windows
// HKCR = Schlüssel des aktuellen Users
var RegistrySchluessel1 = "HKCR\\Directory\\shell\\" + Script_RegistryKennung + "\\";
var RegistrySchluessel2 = "HKCR\\Drive\\shell\\" + Script_RegistryKennung + "\\";

// +++++ WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_ScriptDateiname;
var WinScriptHost_ScriptAufruf_ParameterListe;
var WinScriptHost_ScriptAufruf_ParameterAnzahl;

// +++++ sonstiges
var Parameter;

// ##### Funktionen

function Registry_SchluesselSuchen()
// sucht in der Registry die Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// liefert nur dann true, wenn beide Schlüssel in der Registry vorhanden sind
{
// Annahme: Schlüssel ist in der Registry vorhanden
var Gefunden1 = true; // RegistrySchluessel1 vorhanden
var Gefunden2 = true; // RegistrySchluessel2 vorhanden

// Schlüssel in Registry suchen
// nicht gefunden wird als Laufzeitfehler abgefangen !

// RegistrySchluessel1 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel1); }
catch(Nichtgefunden){ Gefunden1 = false; }

// RegistrySchluessel2 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel2); }
catch(Nichtgefunden){ Gefunden2 = false; }

// true, wenn beide Schlüssel vorhanden sind
return (Gefunden1 && Gefunden2);
}

function Registry_ScriptRegistratur(Flag)
// Flag true, so registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry erzeugen
// false, so de-registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry löschen
{
// +++++ per JScript das Dateisystem referenzieren
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// +++++ weitere Schlüssel für Registratur definieren
var RegistrySchluessel3 = RegistrySchluessel1 + 'command\\';
var RegistrySchluessel4 = RegistrySchluessel2 + 'command\\';
var RegistryWert3 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\\\"';
var RegistryWert4 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\"';

// +++++ Meldungstexte erzeugen
var Kette1 = WScript.ScriptFullName + " wurde";
var Kette2 = "! Im Kontextmenü aller Ordner und Laufwerke steht nun der Menüpunkt \"
+ KontextMenuEintragText + \"";
var Kette3 = "";

// +++++ Registration je nach Art
if (Flag)
{
// registrieren
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel1,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel3,RegistryWert3);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel2,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel4,RegistryWert4);

```



```

}
else
{
    // de-registrieren
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel3);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel1);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel4);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel2);

    Kette1 = Kette1 + "de-";
    Kette3 = " nicht mehr";
}

// +++++ WHS-Analogon zur JScript-Funktion alert()
WScript.Echo(Kette1 + " installiert" + Kette2 + Kette3 + " bereit !");
}

#####

// +++++ WHS-Laufzeitumgebung erzeugen
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// +++++ Dateiname der JS-Datei holen, in der dieses Script steht
var WinScriptHost_ScriptDateiname = WScript.ScriptFullName; // Dateiname der JS-Datei

// +++++ Parameter zum Script-Aufruf holen
var WinScriptHost_ScriptAufruf_ParameterListe = WScript.Arguments;
var WinScriptHost_ScriptAufruf_ParameterAnzahl = WinScriptHost_ScriptAufruf_ParameterListe.length;

// +++++ Registry-Schlüssel in der Registry suchen, also prüfen, ob das Script bereits
//          registriert ist oder nicht
//          gesucht werden Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel1
if (Registry_SchlüsselSuchen())
{
    // beide Schlüssel vorhanden (RegistrySchluessel1 und RegistrySchluessel2 gefunden)
    // Script ist bereits registriert

    // +++++ auf Anzahl der Parameter bei Script-Aufruf prüfen
    if (WinScriptHost_ScriptAufruf_ParameterAnzahl == 0)
    {
        // kein Parameter, also Script aus Registry austragen und Kontextmenü-Eintrag löschen
        Registry_ScriptRegistrierung(false);
    }
    else
    {
        // +++++ mindestens 1 Parameter bei Scriptaufruf
        // aber nur den 1. Parameter holen, also Pfad des Ordners, in dem ein neuer Unterordner erzeugt werden soll
        Parameter = WinScriptHost_ScriptAufruf_ParameterListe(0);

        // prüfen ob Ordner real vorhanden ist
        if (JScript_DateiSystem.FolderExists(Parameter))
        {
            // Ordner real vorhanden
            // also Pfad des Ordners erweitern um den Namen des neuen Unterordners laut StandardOrdnerName
            Parameter = Parameter + StandardOrdnerName;

            // prüfen ob Pfad MIT neuem Unterordner NICHT real vorhanden ist
            if (!JScript_DateiSystem.FolderExists(Parameter))
            {
                // neuer Unterordner ist NICHT real vorhanden, also erzeugen
                JScript_DateiSystem.CreateFolder(Parameter);

                // und den neuen Unterordner als Fenster anzeigen
                WinScriptHost_LaufzeitUmgebung.Run("explorer.exe /select," + Parameter);

                // 1,3 Sekunden warten
                WScript.Sleep(1300);

                // Ordnername editierbar machen durch Simulation des Tastendruckes F2
                WinScriptHost_LaufzeitUmgebung.SendKeys("{F2}");
            }
        }
    }
}

```



```

else
{
// +++++ mindesten 1 Schlüssel fehlt, also Script in Registry eintragen
Registry_ScriptRegistratur(true);
}

```

5.2.3.2.6.3. **Beispiel: Ordner erzeugen**

Erzeugung eines Ordners per Kontextmenü des Windows Explorers.

Nachfolgender Quellcode muss in einer eigenständigen JS-Datei untergebracht sein.

Grund: Das Script muss in der Registry registriert werden, damit es per Kontextmenü aktivierbar ist.

Mit der Script-Registrierung wird in das Kontextmenü der Fenster des Windows Explorers einen Eintrag eingefügt, mit dem ein neuer Ordner erzeugt UND angezeigt werden kann.

Standardname des erzeugten Ordners liegt in der Variablen StandardOrdnerName.

Der Kontextmenü-Eintrag-Text liegt in der Variablen KontextMenuEintragText.

Die Erzeugung des neuen Ordners erfolgt durch Aufruf dieses Scriptes, das dazu BEREITS in der Registry registriert sein muss.

Mit der Script-De-Registrierung wird zugleich der Kontextmenü-Eintrag entfernt.

Scriptaufruf: script_js_date [ordner_name]

```

ordner_name: optional
    Name des Ordners in dem ein Unterordner erzeugt werden soll
    wenn nicht kodiert, so gilt:
        Ist das Script noch nicht in der Registry registriert, so wird es registriert.
        Ist ein Teil der Registry-Schlüssel zum Script nicht vorhanden, so wird
        das Script neu registriert.
        Ist das Script bereits in der Registry registriert, so wird es de-registriert.

```

// ##### Variablen

```

// +++++ Variablen, die frei wählbar sind
var StandardOrdnerName = "Neuer Ordner";
var KontextMenuEintragText = "Neuen Unterordner erzeugen";
var Script_RegistryKennung = "TestScript";

```

// +++++ interne Schlüssel des Script zum Registrieren und De-Registrieren (nicht verändern)

```

// Lage der Schlüssel in der Registry:
// Directory und Drive für Dateisystem-Verwaltung durch Windows
// HKCR = Schlüssel des aktuellen Users
var RegistrySchlüssel1 = "HKCR\\Directory\\shell\\" + Script_RegistryKennung + "\\";
var RegistrySchlüssel2 = "HKCR\\Drive\\shell\\" + Script_RegistryKennung + "\\";

```

```

// +++++ WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_ScriptDateiname;
var WinScriptHost_ScriptAufruf_ParameterListe;
var WinScriptHost_ScriptAufruf_ParameterAnzahl;

```

```

// +++++ sonstiges
var Parameter;

```

// ##### Funktionen

```

function Registry_SchlüsselSuchen()
// sucht in der Registry die Schlüssel laut Variablen RegistrySchlüssel1 und RegistrySchlüssel2
// liefert nur dann true, wenn beide Schlüssel in der Registry vorhanden sind
{
// Annahme: Schlüssel ist in der Registry vorhanden
var Gefunden1 = true; // RegistrySchlüssel1 vorhanden
var Gefunden2 = true; // RegistrySchlüssel2 vorhanden

// Schlüssel in Registry suchen
// nicht gefunden wird als Laufzeitfehler abgefangen !

// RegistrySchlüssel1 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchlüssel1); }
catch(Nichtgefunden){ Gefunden1 = false; }

```



```

// RegistrySchluessel2 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel2); }
catch(Nichtgefunden){ Gefunden2 = false; }

// true, wenn beide Schlüssel vorhanden sind
return (Gefunden1 && Gefunden2);
}

function Registry_ScriptRegistratur(Flag)
// Flag true, so registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
//      in der Registry erzeugen
//      false, so de-registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
//      in der Registry löschen
{
// +++++ per JScript das Dateisystem referenzieren
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// +++++ weitere Schlüssel für Registratur definieren
var RegistrySchluessel3 = RegistrySchluessel1 + 'command\\';
var RegistrySchluessel4 = RegistrySchluessel2 + 'command\\';
var RegistryWert3 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\' "%L\\\\"';
var RegistryWert4 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\' "%L\\"';

// +++++ Meldungstexte erzeugen
var Kette1 = WScript.ScriptFullName + " wurde";
var Kette2 = "! Im Kontextmenü aller Ordner und Laufwerke steht nun der Menüpunkt \"
      + KontextMenuEintragText + \"";
var Kette3 = "";

// +++++ Registration je nach Art
if (Flag)
{
// registrieren
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel1,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel3,RegistryWert3);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel2,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel4,RegistryWert4);
}
else
{
// de-registrieren
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel3);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel1);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel4);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel2);

Kette1 = Kette1 + "de-";
Kette3 = " nicht mehr";
}

// +++++ WHS-Analogon zur JScript-Funktion alert()
WScript.Echo(Kette1 + " installiert" + Kette2 + Kette3 + " bereit !");
}

#####

// +++++ WHS-Laufzeitumgebung erzeugen
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// +++++ Dateiname der JS-Datei holen, in der dieses Script steht
var WinScriptHost_ScriptDateiname = WScript.ScriptFullName; // Dateiname der JS-Datei

// +++++ Parameter zum Script-Aufruf holen
var WinScriptHost_ScriptAufruf_ParameterListe = WScript.Arguments;
var WinScriptHost_ScriptAufruf_ParameterAnzahl = WinScriptHost_ScriptAufruf_ParameterListe.length;

// +++++ Registry-Schlüssel in der Registry suchen, also prüfen, ob das Script bereits
//      registriert ist oder nicht
//      gesucht werden Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel1
if (Registry_SchluesselSuchen())
{
// beide Schlüssel vorhanden (RegistrySchluessel1 und RegistrySchluessel2 gefunden)
// Script ist bereits registriert

```



```
// +++++ auf Anzahl der Parameter bei Script-Aufruf prüfen
if (WinScriptHost_ScriptAufruf_ParameterAnzahl == 0)
{
    // kein Parameter, also Script aus Registry austragen und Kontextmenü-Eintrag löschen
    Registry_ScriptRegistratur(false);
}
else
{
    // +++++ mindestens 1 Parameter bei Scriptaufruf
    // aber nur den 1. Parameter holen, also Pfad des Ordners, in dem ein neuer Unterordner erzeugt werden soll
    Parameter = WinScriptHost_ScriptAufruf_ParameterListe(0);

    // prüfen ob Ordner real vorhanden ist
    if (JScript_DateiSystem.FolderExists(Parameter))
    {
        // Ordner real vorhanden
        // also Pfad des Ordners erweitern um den Namen des neuen Unterordners laut StandardOrdnerName
        Parameter = Parameter + StandardOrdnerName;

        // prüfen ob Pfad MIT neuem Unterordner NICHT real vorhanden ist
        if (!JScript_DateiSystem.FolderExists(Parameter))
        {
            // neuer Unterordner ist NICHT real vorhanden, also erzeugen
            JScript_DateiSystem.CreateFolder(Parameter);

            // und den neuen Unterordner als Fenster anzeigen
            WinScriptHost_LaufzeitUmgebung.Run("explorer.exe /select," + Parameter);

            // 1,3 Sekunden warten
            WScript.Sleep(1300);

            // Ordnername editierbar machen durch Simulation des Tastendruckes F2
            WinScriptHost_LaufzeitUmgebung.SendKeys("{F2}");
        }
    }
    else
    {
        // +++++ mindesten 1 Schlüssel fehlt, also Script in Registry eintragen
        Registry_ScriptRegistratur(true);
    }
}
```

5.2.3.2.6.4. **Beispiel: Lokales Programm starten**

Windows Explorer als lokales Programm starten

Nachfolgender Quellcode muss in einer eigenständigen JS-Datei untergebracht sein.

Grund: Das Script muss in der Registry registriert werden, damit es per Kontextmenü aktivierbar ist.

Mit der Script-Registrierung wird in das Kontextmenü der Fenster des Windows Explorers einen Eintrag eingefügt, mit dem ein neuer Ordner erzeugt UND angezeigt werden kann.

Standardname des erzeugten Ordners liegt in der Variablen StandardOrdnerName.

Der Kontextmenü-Eintrag-Text liegt in der Variablen KontextMenuEintragText.

Die Erzeugung des neuen Ordners erfolgt durch Aufruf dieses Scriptes, das dazu BEREITS in der Registry registriert sein muss.

Mit der Script-De-Registrierung wird zugleich der Kontextmenü-Eintrag entfernt.

Scriptaufruf: script_js_date [ordner_name]

ordner_name: optional
 Name des Ordners in dem ein Unterordner erzeugt werden soll
 wenn nicht kodiert, so gilt:
 Ist das Script noch nicht in der Registry registriert, so wird es registriert.
 Ist ein Teil der Registry-Schlüssel zum Script nicht vorhanden, so wird das Script neu registriert.
 Ist das Script bereits in der Registry registriert, so wird es de-registriert.

// ##### Variablen



```
// +++++ Variablen, die frei wählbar sind
var StandardOrdnerName = "Neuer Ordner";
var KontextMenuEintragText = "Neuen Unterordner erzeugen";
var Script_RegistryKennung = "TestScript";

// +++++ interne Schlüssel des Script zum Registrieren und De-Registrieren (nicht verändern)
// Lage der Schlüssel in der Registry:
// Directory und Drvie für Dateisystem-Verwaltung durch Windows
// HKCR = Schlüssel des aktuellen Users
var RegistrySchluessel1 = "HKCR\\Directory\\shell\\" + Script_RegistryKennung + "\\";
var RegistrySchluessel2 = "HKCR\\Drive\\shell\\" + Script_RegistryKennung + "\\";

// +++++ WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_ScriptDateiname;
var WinScriptHost_ScriptAufruf_ParameterListe;
var WinScriptHost_ScriptAufruf_ParameterAnzahl;

// +++++ sonstiges
var Parameter;

// ##### Funktionen

function Registry_SchluesselSuchen()
// sucht in der Registry die Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// liefert nur dann true, wenn beide Schlüssel in der Registry vorhanden sind
{
// Annahme: Schlüssel ist in der Registry vorhanden
var Gefunden1 = true; // RegistrySchluessel1 vorhanden
var Gefunden2 = true; // RegistrySchluessel2 vorhanden

// Schlüssel in Registry suchen
// nicht gefunden wird als Laufzeitfehler abgefangen !

// RegistrySchluessel1 suchen
try {WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel1);}
catch(Nichtgefunden){Gefunden1 = false;}

// RegistrySchluessel2 suchen
try {WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel2);}
catch(Nichtgefunden){Gefunden2 = false;}

// true, wenn beide Schlüssel vorhanden sind
return (Gefunden1 && Gefunden2);
}

function Registry_ScriptRegistratur(Flag)
// Flag true, so registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry erzeugen
// false, so de-registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry löschen
{
// +++++ per JScript das Dateisystem referenzieren
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// +++++ weitere Schlüssel für Registratur definieren
var RegistrySchluessel3 = RegistrySchluessel1 + 'command\\';
var RegistrySchluessel4 = RegistrySchluessel2 + 'command\\';
var RegistryWert3 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\\\"';
var RegistryWert4 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\\\"';

// +++++ Meldungstexte erzeugen
var Kette1 = WScript.ScriptFullName + " wurde";
var Kette2 = "! \\Im Kontextmenü aller Ordner und Laufwerke steht nun der Menüpunkt \"
+ KontextMenuEintragText + \"\"";
var Kette3 = "";

// +++++ Registration je nach Art
if (Flag)
{
// registrieren
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel1,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel3,RegistryWert3);
```



```

WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel2,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel4,RegistryWert4);
}
else
{
// de-registrieren
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel3);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel1);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel4);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel2);

Kette1 = Kette1 + "de-";
Kette3 = " nicht mehr";
}

// +++++ WHS-Analogon zur JScript-Funktion alert()
WScript.Echo(Kette1 + " installiert" + Kette2 + Kette3 + " bereit !");
}

// #####

// +++++ WHS-Laufzeitumgebung erzeugen
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// +++++ Dateiname der JS-Datei holen, in der dieses Script steht
var WinScriptHost_ScriptDateiname = WScript.ScriptFullName; // Dateiname der JS-Datei

// +++++ Parameter zum Script-Aufruf holen
var WinScriptHost_ScriptAufruf_ParameterListe = WScript.Arguments;
var WinScriptHost_ScriptAufruf_ParameterAnzahl = WinScriptHost_ScriptAufruf_ParameterListe.length;

// +++++ Registry-Schlüssel in der Registry suchen, also prüfen, ob das Script bereits
// registriert ist oder nicht
// gesucht werden Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel1
if (Registry_SchluesselSuchen())
{
// beide Schlüssel vorhanden (RegistrySchluessel1 und RegistrySchluessel2 gefunden)
// Script ist bereits registriert

// +++++ auf Anzahl der Parameter bei Script-Aufruf prüfen
if (WinScriptHost_ScriptAufruf_ParameterAnzahl == 0)
{
// kein Parameter, also Script aus Registry austragen und Kontextmenü-Eintrag löschen
Registry_ScriptRegistratur(false);
}
else
{
// +++++ mindestens 1 Parameter bei Scriptaufruf
// aber nur den 1. Parameter holen, also Pfad des Ordners, in dem ein neuer Unterordner erzeugt werden soll
Parameter = WinScriptHost_ScriptAufruf_ParameterListe(0);

// prüfen ob Ordner real vorhanden ist
if (JScript_DateiSystem.FolderExists(Parameter))
{
// Ordner real vorhanden
// also Pfad des Ordners erweitern um den Namen des neuen Unterordners laut StandardOrdnerName
Parameter = Parameter + StandardOrdnerName;

// prüfen ob Pfad MIT neuem Unterordner NICHT real vorhanden ist
if (!JScript_DateiSystem.FolderExists(Parameter))
{
// neuer Unterordner ist NICHT real vorhanden, also erzeugen
JScript_DateiSystem.CreateFolder(Parameter);

// und den neuen Unterordner als Fenster anzeigen
WinScriptHost_LaufzeitUmgebung.Run("explorer.exe /select," + Parameter);

// 1,3 Sekunden warten
WScript.Sleep(1300);

// Ordnername editierbar machen durch Simulation des Tastendruckes F2
WinScriptHost_LaufzeitUmgebung.SendKeys("{F2}");
}
}
}
}

```




```

}
}
else
{
// +++++ mindesten 1 Schlüssel fehlt, also Script in Registry eintragen
Registry_ScriptRegistratur(true);
}

```

5.2.3.2.6.5. **Beispiel: Tastensimulation**

Tastensimulation für den Windows Explorer

Nachfolgender Quellcode muss in einer eigenständigen JS-Datei untergebracht sein.

Grund: Das Script muss in der Registry registriert werden, damit es per Kontextmenü aktivierbar ist.

Mit der Script-Registrierung wird in das Kontextmenü der Fenster des Windows Explorers einen Eintrag eingefügt, mit dem ein neuer Ordner erzeugt UND angezeigt werden kann.

Standardname des erzeugten Ordners liegt in der Variablen StandardOrdnerName.

Der Kontextmenü-Eintrag-Text liegt in der Variablen KontextMenuEintragText.

Die Erzeugung des neuen Ordners erfolgt durch Aufruf dieses Scriptes, das dazu BEREITS in der Registry registriert sein muss.

Mit der Script-De-Registrierung wird zugleich der Kontextmenü-Eintrag entfernt.

Scriptaufruf: script_js_date [ordner_name]

ordner_name: optional
 Name des Ordners in dem ein Unterordner erzeugt werden soll
 wenn nicht kodiert, so gilt:
 Ist das Script noch nicht in der Registry registriert, so wird es registriert.
 Ist ein Teil der Registry-Schlüssel zum Script nicht vorhanden, so wird das Script neu registriert.
 Ist das Script bereits in der Registry registriert, so wird es de-registriert.

// ##### Variablen

```

// +++++ Variablen, die frei wählbar sind
var StandardOrdnerName = "Neuer Ordner";
var KontextMenuEintragText = "Neuen Unterordner erzeugen";
var Script_RegistryKennung = "TestScript";

```

// +++++ interne Schlüssel des Script zum Registrieren und De-Registrieren (nicht verändern)

```

// Lage der Schlüssel in der Registry:
// Directory und Drive für Dateisystem-Verwaltung durch Windows
// HKCR = Schlüssel des aktuellen Users
var RegistrySchluessel1 = "HKCR\\Directory\\shell\\" + Script_RegistryKennung + "\\";
var RegistrySchluessel2 = "HKCR\\Drive\\shell\\" + Script_RegistryKennung + "\\";

```

```

// +++++ WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_ScriptDateiname;
var WinScriptHost_ScriptAufruf_ParameterListe;
var WinScriptHost_ScriptAufruf_ParameterAnzahl;

```

```

// +++++ sonstiges
var Parameter;

```

// ##### Funktionen

```

function Registry_SchluesselSuchen()
// sucht in der Registry die Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// liefert nur dann true, wenn beide Schlüssel in der Registry vorhanden sind
{
// Annahme: Schlüssel ist in der Registry vorhanden
var Gefunden1 = true; // RegistrySchluessel1 vorhanden
var Gefunden2 = true; // RegistrySchluessel2 vorhanden

```

```

// Schlüssel in Registry suchen
// nicht gefunden wird als Laufzeitfehler abgefangen !

```

```

// RegistrySchluessel1 suchen

```



```

try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel1);}
catch(Nichtgefunden){Gefunden1 = false;}

// RegistrySchluessel2 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel2);}
catch(Nichtgefunden){Gefunden2 = false;}

// true, wenn beide Schlüssel vorhanden sind
return (Gefunden1 && Gefunden2);
}

function Registry_ScriptRegistratur(Flag)
// Flag true, so registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
//      in der Registry erzeugen
//      false, so de-registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
//      in der Registry löschen
{
// +++++ per JScript das Dateisystem referenzieren
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// +++++ weitere Schlüssel für Registratur definieren
var RegistrySchluessel3 = RegistrySchluessel1 + 'command\\';
var RegistrySchluessel4 = RegistrySchluessel2 + 'command\\';
var RegistryWert3 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\' "%L\\\\"';
var RegistryWert4 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\' "%L\\\\"';

// +++++ Meldungstexte erzeugen
var Kette1 = WScript.ScriptFullName + " wurde";
var Kette2 = "! \Im Kontextmenü aller Ordner und Laufwerke steht nun der Menüpunkt \"
      + KontextMenuEintragText + \"\"";
var Kette3 = "";

// +++++ Registration je nach Art
if (Flag)
{
// registrieren
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel1,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel3,RegistryWert3);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel2,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel4,RegistryWert4);
}
else
{
// de-registrieren
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel3);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel1);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel4);
WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel2);

Kette1 = Kette1 + "de-";
Kette3 = " nicht mehr";
}

// +++++ WHS-Analogon zur JScript-Funktion alert()
WScript.Echo(Kette1 + " installiert" + Kette2 + Kette3 + " bereit !");
}

// #####

// +++++ WHS-Laufzeitumgebung erzeugen
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// +++++ Dateiname der JS-Datei holen, in der dieses Script steht
var WinScriptHost_ScriptDateiname = WScript.ScriptFullName; // Dateiname der JS-Datei

// +++++ Parameter zum Script-Aufruf holen
var WinScriptHost_ScriptAufruf_ParameterListe = WScript.Arguments;
var WinScriptHost_ScriptAufruf_ParameterAnzahl = WinScriptHost_ScriptAufruf_ParameterListe.length;

// +++++ Registry-Schlüssel in der Registry suchen, also prüfen, ob das Script bereits
//      registriert ist oder nicht
//      gesucht werden Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel1
if (Registry_SchluesselSuchen())
{

```



```
// beide Schlüssel vorhanden (RegistrySchlüssel1 und RegistrySchlüssel2 gefunden)
// Script ist bereits registriert

// +++++ auf Anzahl der Parameter bei Script-Aufruf prüfen
if (WinScriptHost_ScriptAufruf_ParameterAnzahl == 0)
{
    // kein Parameter, also Script aus Registry austragen und Kontextmenü-Eintrag löschen
    Registry_ScriptRegistratur(false);
}
else
{
    // +++++ mindestens 1 Parameter bei Scriptaufruf
    // aber nur den 1. Parameter holen, also Pfad des Ordners, in dem ein neuer Unterordner erzeugt werden soll
    Parameter = WinScriptHost_ScriptAufruf_ParameterListe(0);

    // prüfen ob Ordner real vorhanden ist
    if (JScript_DateiSystem.FolderExists(Parameter))
    {
        // Ordner real vorhanden
        // also Pfad des Ordners erweitern um den Namen des neuen Unterordners laut StandardOrdnerName
        Parameter = Parameter + StandardOrdnerName;

        // prüfen ob Pfad MIT neuem Unterordner NICHT real vorhanden ist
        if (!JScript_DateiSystem.FolderExists(Parameter))
        {
            // neuer Unterordner ist NICHT real vorhanden, also erzeugen
            JScript_DateiSystem.CreateFolder(Parameter);

            // und den neuen Unterordner als Fenster anzeigen
            WinScriptHost_LaufzeitUmgebung.Run("explorer.exe /select," + Parameter);

            // 1,3 Sekunden warten
            WScript.Sleep(1300);

            // Ordnername editierbar machen durch Simulation des Tastendruckes F2
            WinScriptHost_LaufzeitUmgebung.SendKeys("{F2}");
        }
    }
    else
    {
        // +++++ mindesten 1 Schlüssel fehlt, also Script in Registry eintragen
        Registry_ScriptRegistratur(true);
    }
}
```

5.2.3.2.6.6. **Beispiel: Zugriff auf Kontextmenü des Windows Explorers**

Kontextmenü-Eintrag für den Windows Explorer erzeugen

Nachfolgender Quellcode muss in einer eigenständigen JS-Datei untergebracht sein.

Grund: Das Script muss in der Registry registriert werden, damit es per Kontextmenü aktivierbar ist.

Mit der Script-Registrierung wird in das Kontextmenü der Fenster des Windows Explorers ein Eintrag eingefügt, mit dem ein neuer Ordner erzeugt UND angezeigt werden kann.

Standardname des erzeugten Ordners liegt in der Variablen StandardOrdnerName.

Der Kontextmenü-Eintrag-Text liegt in der Variablen KontextMenuEintragText.

Die Erzeugung des neuen Ordners erfolgt durch Aufruf dieses Scriptes, das dazu BEREITS in der Registry registriert sein muss.

Mit der Script-De-Registrierung wird zugleich der Kontextmenü-Eintrag entfernt.

Scriptaufruf: script_js_date [ordner_name]

ordner_name: optional

Name des Ordners in dem ein Unterordner erzeugt werden soll
wenn nicht kodiert, so gilt:

Ist das Script noch nicht in der Registry registriert, so wird es registriert.

Ist ein Teil der Registry-Schlüssel zum Script nicht vorhanden, so wird das Script neu registriert.

Ist das Script bereits in der Registry registriert, so wird es de-registriert.



```
// ##### Variablen

// +++++ Variablen, die frei wählbar sind
var StandardOrdnerName = "Neuer Ordner";
var KontextMenuEintragText = "Neuen Unterordner erzeugen";
var Script_RegistryKennung = "TestScript";

// +++++ interne Schlüssel des Script zum Registrieren und De-Registrieren (nicht verändern)
// Lage der Schlüssel in der Registry:
// Directory und Drvie für Dateisystem-Verwaltung durch Windows
// HKCR = Schlüssel des aktuellen Users
var RegistrySchluessel1 = "HKCR\\Directory\\shell\\" + Script_RegistryKennung + "\\";
var RegistrySchluessel2 = "HKCR\\Drive\\shell\\" + Script_RegistryKennung + "\\";

// +++++ WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung;
var WinScriptHost_ScriptDateiname;
var WinScriptHost_ScriptAufruf_ParameterListe;
var WinScriptHost_ScriptAufruf_ParameterAnzahl;

// +++++ sonstiges
var Parameter;

// ##### Funktionen

function Registry_SchluesselSuchen()
// sucht in der Registry die Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// liefert nur dann true, wenn beide Schlüssel in der Registry vorhanden sind
{
// Annahme: Schlüssel ist in der Registry vorhanden
var Gefunden1 = true; // RegistrySchluessel1 vorhanden
var Gefunden2 = true; // RegistrySchluessel2 vorhanden

// Schlüssel in Registry suchen
// nicht gefunden wird als Laufzeitfehler abgefangen !

// RegistrySchluessel1 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel1); }
catch(Nichtgefunden){ Gefunden1 = false; }

// RegistrySchluessel2 suchen
try { WinScriptHost_LaufzeitUmgebung.RegRead(RegistrySchluessel2); }
catch(Nichtgefunden){ Gefunden2 = false; }

// true, wenn beide Schlüssel vorhanden sind
return (Gefunden1 && Gefunden2);
}

function Registry_ScriptRegistratur(Flag)
// Flag true, so registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry erzeugen
// false, so de-registrieren, also Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel2
// in der Registry löschen
{
// +++++ per JScript das Dateisystem referenzieren
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

// +++++ weitere Schlüssel für Registratur definieren
var RegistrySchluessel3 = RegistrySchluessel1 + 'command\\';
var RegistrySchluessel4 = RegistrySchluessel2 + 'command\\';
var RegistryWert3 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\\\"';
var RegistryWert4 = 'wscript.exe \'' + WinScriptHost_ScriptDateiname + '\" \"%L\\\"';

// +++++ Meldungstexte erzeugen
var Kette1 = WScript.ScriptFullName + " wurde";
var Kette2 = '! \\Im Kontextmenü aller Ordner und Laufwerke steht nun der Menüpunkt \''
+ KontextMenuEintragText + '\"';
var Kette3 = "";

// +++++ Registration je nach Art
if (Flag)
{
// registrieren
```



```

WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel1,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel3,RegistryWert3);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel2,KontextMenuEintragText);
WinScriptHost_LaufzeitUmgebung.RegWrite(RegistrySchluessel4,RegistryWert4);
}
else
{
    // de-registrieren
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel3);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel1);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel4);
    WinScriptHost_LaufzeitUmgebung.RegDelete(RegistrySchluessel2);

    Kette1 = Kette1 + "de-";
    Kette3 = " nicht mehr";
}

// +++++ WHS-Analogon zur JScript-Funktion alert()
WScript.Echo(Kette1 + " installiert" + Kette2 + Kette3 + " bereit !");
}

// #####

// +++++ WHS-Laufzeitumgebung erzeugen
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");

// +++++ Dateiname der JS-Datei holen, in der dieses Script steht
var WinScriptHost_ScriptDateiname = WScript.ScriptFullName; // Dateiname der JS-Datei

// +++++ Parameter zum Script-Aufruf holen
var WinScriptHost_ScriptAufruf_ParameterListe = WScript.Arguments;
var WinScriptHost_ScriptAufruf_ParameterAnzahl = WinScriptHost_ScriptAufruf_ParameterListe.length;

// +++++ Registry-Schlüssel in der Registry suchen, also prüfen, ob das Script bereits
//          registriert ist oder nicht
//          gesucht werden Schlüssel laut Variablen RegistrySchluessel1 und RegistrySchluessel1
if (Registry_SchluesselSuchen())
{
    // beide Schlüssel vorhanden (RegistrySchluessel1 und RegistrySchluessel2 gefunden)
    // Script ist bereits registriert

    // +++++ auf Anzahl der Parameter bei Script-Aufruf prüfen
    if (WinScriptHost_ScriptAufruf_ParameterAnzahl == 0)
    {
        // kein Parameter, also Script aus Registry austragen und Kontextmenü-Eintrag löschen
        Registry_ScriptRegistrierung(false);
    }
    else
    {
        // +++++ mindestens 1 Parameter bei Scriptaufruf
        // aber nur den 1. Parameter holen, also Pfad des Ordners, in dem ein neuer Unterordner erzeugt werden soll
        Parameter = WinScriptHost_ScriptAufruf_ParameterListe(0);

        // prüfen ob Ordner real vorhanden ist
        if (JScript_DateiSystem.FolderExists(Parameter))
        {
            // Ordner real vorhanden
            // also Pfad des Ordners erweitern um den Namen des neuen Unterordners laut StandardOrdnerName
            Parameter = Parameter + StandardOrdnerName;

            // prüfen ob Pfad MIT neuem Unterordner NICHT real vorhanden ist
            if (!JScript_DateiSystem.FolderExists(Parameter))
            {
                // neuer Unterordner ist NICHT real vorhanden, also erzeugen
                JScript_DateiSystem.CreateFolder(Parameter);

                // und den neuen Unterordner als Fenster anzeigen
                WinScriptHost_LaufzeitUmgebung.Run("explorer.exe /select," + Parameter);

                // 1,3 Sekunden warten
                WScript.Sleep(1300);

                // Ordnername editierbar machen durch Simulation des Tastendruckes F2
                WinScriptHost_LaufzeitUmgebung.SendKeys("{F2}");
            }
        }
    }
}

```



```

    }
  }
}
else
{
  // +++++ mindesten 1 Schlüssel fehlt, also Script in Registry eintragen
  Registry_ScriptRegistratur(true);
}

```

5.2.3.2.6.7. **Beispiel: Zugriff auf PATH-Variable**

Zugriff auf PATH-Variable

```

// WinScriptHost-Objekte
var WinScriptHost_LaufzeitUmgebung = WScript.CreateObject("WScript.Shell");
var WinScriptHost_PATHVariable = WinScriptHost_LaufzeitUmgebung.ExpandEnvironmentStrings("%path%");

// JScript-Dateisystem
var JScript_DateiSystem = new ActiveXObject("Scripting.FileSystemObject");

var PfadTrenner = ",";
var TeilPfad;
var TeilPfade_Anzahl = 0; // ab 1
var TeilPfade_Ungueltige = "";

var Zahler = 0,

// "\" ersetzen durch Leerkette
WinScriptHost_PATHVariable = WinScriptHost_PATHVariable.replace(/\\/g, "");

// Semikolon zählen (Pfad-Trenner) durch zeichenweises abklappern
for (Zahler = 0; Zahler < WinScriptHost_PATHVariable.length; Zahler++)
{
  if (WinScriptHost_PATHVariable.charAt(Zahler) == PfadTrenner) { TeilPfade_Anzahl++; }
}

// Feld der Teilpfade erzeugen
var TeilPfade_Feld = new Array(TeilPfade_Anzahl-1); // Index ab 0

// Path-Kette nach Feld: Feldelement liegt zwischen Pfadtrenner
TeilPfade_Feld = WinScriptHost_PATHVariable.split(PfadTrenner);

// Teilpfad auf realen Ordner prüfen
for (Zahler = 0; Zahler < TeilPfade_Feld.length; Zahler++)
{
  // nächster Teilpfad
  TeilPfad=TeilPfade_Feld[Zahler];

  // prüfen auf realen Ordner
  if (JScript_DateiSystem.FolderExists(TeilPfad) == false)
  {
    // kein realer Ordner vorhanden
    TeilPfade_Ungueltige = TeilPfade_Ungueltige + TeilPfad + "\\n";
  }

  // Anzeige aller ungültigen Teilpfade im Path
  if (TeilPfade_Ungueltige == "")
  { WScript.Echo("Teilpfade existieren real !"); }
  else
  WScript.Echo("Ungültige Teilpfade sind \\n\\n" + TeilPfade_Ungueltige);
}

```

5.2.4. **Windows Media Player 7.1 und Internet Explorer**

Der Windows Media Player dient als Addon u.a. zur Wiedergabe diverser Medien anhand von Medien-Dateien oder einer Playliste von Medien-Dateien.

Der Mediaplayer 7.1 wurde exemplarisch gewählt, da im SDK die Scriptprogrammierung offeriert wurde.

Es besteht die Möglichkeit, dass Nachfolgeversionen des Mediaplayer 7.1 ebenfalls diese Scriptprogrammierung unterstützen.

Vermutlich wurde aber ab MediaPlayer 10 Inkompatibilität hergestellt: Man muss es eben austesten.

Der Windows Media Player ist im HTML-Dokument per HTML und optional per JScript ansteuerbar, wobei der Programmierungsaufwand in JScript erheblich ist. Eine sehr einfache, aber wenig dynamische Programmierungsvariante ist beim Objekt bgound zu finden (siehe dort).

Der Windows Media Player 7.1 wird für den Netscape im Gegensatz zum Windows Media Player 6.4 als Plugin **nicht** mehr unterstützt. Ab dem IE 6.x werden generell keine Plugins mehr unterstützt. Microsoft favorisiert ihr eigenes ActiveX-Konzept und überlässt z.T.

Fremdherstellern die Programmierung von ActiveX-Controls z.B. als Addon zum Internet Explorer. Vorteil ist die Normung per Active-X-



Schnittstelle und somit die Kompletintegration z.B. des Players und IE in Windows. Nachteil ist das Ausbieten von Konkurrenzprodukten, wenn Microsoft die notwendigen Schnittstellen nicht komplett offengelegt hat und damit eine Weiterentwicklung durch die Konkurrenz vorallem für Anwender erschwert, die nicht unbedingt mit der Microsoft-Player-Software arbeiten wollen bzw. andere Features erwarten, also sie Microsoft bisher implementiert hat.

Eine etwas veraltete und ebenfalls programmtechnisch aufwändige Alternative ist die Medienwiedergabe per Microsoft Direct Animation (DA) als einer der Vorgänger bzw. Komponenten des Windows Media Player.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit Kompression genutzt wird z.B. bei onlick-Handler auf IMG klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhatten Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show()erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz



Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per `.show()` erzeugt, bricht der Browser das Dokument mit `.show()` ab (Scriptfehler).

Der Popublocker für die leere Seite verursacht den Programmfehler im Dokument mit `.show()`. Es wird folgende Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popublocker hat ein Popupfenster geblockt. Sie können den Popublocker deaktivieren oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popublocker einschalten
weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popublocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popublocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popublocker verwaltet wird.

Der Popublocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popublocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popublockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus
onblur
onfocusin
onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popublockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

`window.focus();`

`window.document.focus();`

`if(document.body!=null)`

`{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)`

`{document.body.focus();}`

`}`

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

`popupzeiger.show(...);`

Hinweis: Der Popubfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

`windows.focus()` `document.focus()` und `body.focus()` funktionieren NICHT

zwischen Register in einem IE-Fenster

zwischen Fensters z.B. in Taskleiste

Hinweis:

`.focus()` setzt Element aktiv, gibt dem Element den Focus und feuert dann `onfocus`

`.setActive()` ist Teilmenge von `.focus()`: nur das aktiv setzen

funktioniert nicht mit allen Elementen, mit denen `.focus()` funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:

z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.

Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtsstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.

Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.

Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert



wurde.

Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X-Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp (http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind



Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)
http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue



ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: •



<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>

Aktive Inhalte im Internet Explorer

Ab IE 6.0 ist das Blockieren aktiver Inhalte möglich, z.B. als Standardeinstellung. Es wird also dem IE verboten, JScript zu nutzen. Daher muss mit Start der Webseite auf das Blockieren von Inhalten der Webseite, die auf JScript basieren, aufmerksam gemacht werden. Bleibt die Blockierung aktiv, so muss die Webseite ALLE Elemente, die per Script angesteuert werden, inaktiv machen: Am besten garnicht erst anzeigen. Oder es wird eine scriptfreie Version der Webseite per <NOSCRIPT> aktiviert, wobei dann Browser vorzuziehbar sind, die z.B. CSS

exakter rendern als der IE (will man keine IE-spezifischen HTML-Elemente verwenden).

Wenn der IE 6.x aktive Inhalte blockiert, wird NOSCRIPT-Tag aktiviert, Ausnahme: Frameset

FRAMESET ist ein aktiver Inhalt:

Da der Frameset anstelle <BODY> kodiert sein muss, gilt:

Alle Tags, die für BODY zulässig sind, werden ignoriert, auch NOSCRIPT.

Wird neben Frameset noch BODY kodiert, so wird Frameset ignoriert.

Die Freigabe der Scriptblockierung erzeugt Ausführung aller Script-Teile inklusive der Eventauslösungen

Bsp.: Folgendes funktioniert vom Dokument, das window.open() hat im geöffneten Dokument (Quelltext im Dokument das window.open() verwendet):

```
function Y_unload(X00){X85[X00].close();}

var X85=new Array();var X86=new Array();

X85[0]=window.open(...);

var X87=parent.Y_unload(0); X86[0]=new Function("X87);

X85[0].document.body.onunload=X86[0];
```

Wird die Scriptblockierung im geöffneten Fenster abgeschaltet,
so wird das Fenster geschlossen, weil onunload ausgelöst wird.

Achtung: document.body.onunload funktioniert ev. nicht mehr
wenn z.B. mit attachevent() aktiviert wurde

Folgende Metatags sind für den IE 6.x aktiver Inhalt:

```
<META HTTP-EQUIV="imagetoolbar" CONTENT="no">
    unterdrückt NICHT IE-Kontextmenü rechte Maus auf Bild

<META HTTP-EQUIV="site-enter" CONTENT="revealtrans(duration=0.3, transition=12)">
<META HTTP-EQUIV="site-exit" CONTENT="revealtrans(duration=0.3, transition=12)">
```

Achtung: Für das Hinzufügen von Elementen in den BODY (document.body) per DOM-Funktion createElement() MUSS der Body komplett geparkt sein (document.body.readyState == 'complete').

Grund: Es wird standargemäß immer am Ende des BODY angefügt.

Für das Hinzufügen nicht an das Ende des BODY muss im HTML-Code ein Platzhalter z.B. DIV kodiert sein,
innerhalb

dessen dann die neuen HTML-Elemente erzeugt werden.

Beispiel für Prüfung auf ActiveX beim IE:

```
var ActiveXAktiv = false;

var NavigatorObjekt = window.navigator;           // Zeiger

var BrowserArt      = NavigatorObjekt.userAgent;   // String
var IEerkannt       = (BrowserArt.indexOf('IE') > -1); // true, so IE erkannt

var Plattform       = NavigatorObjekt.platform;     // String
var Win32Erkannt    = (Plattform == "Win32");       // true, so Win32-Bit erkannt

Aktiv               = (IEerkannt && Win32Erkannt);   // true, so IE und Win32-Bit erkannt,
// also ActiveX möglich
```

Beispiel für Browser-Erkennung:

```
var UserAgent        = navigator.userAgent;
var Browser_Version_Haupt = 0;
var Browser_Version_Unter = 0;

// Browserversion ermitteln
```



```

if (UserAgent != "")
{
    // Hauptversion des Browsers holen
    for (var i=0; i < 10; i++)
    {
        eval(      'if (UserAgent.indexOf("'
                    + i
                    + ',") != -1)'
                    + '{Browser_Version_Haupt = '
                    + i
                    + '};}'
                );
    }

    // Unterversion des Browsers holen
    for (var i=0; i < 100; i++)
    {
        eval(      'if (UserAgent.indexOf("'
                    + i
                    + ',") != -1)'
                    + '{Browser_Version_Unter = '
                    + i
                    + '};}'
                );
    }

    // wenn Unterversion des Browsers am Ende eine Null hat, so diese eliminieren z.B. 50 zu 5
    if ((Browser_Version_Unter % 10) == 0)
    {Browser_Version_Unter = Browser_Version_Unter / 10;}    // Bsp.: 50 % 10 = 5 Rest 0, also 5
}

var NSunter6=    (      (!document.all)                // kein NS kennt document.all
                  || (document.layers)                // bis NS unter 6.x
                );

var NSab6=    (      (!document.all)                // kein NS kennt document.all
                  && (document.getElementById)        // ab NS 6.x ist getElementById implementiert
                  // theoretisch wäre zusätzlich (!document.layers) kodierbar, aber mit Implementation
                  // von getElementById wurde gleichzeitig document.layers abgeschafft
                );

var NS4x  =    (      (NSunter6)
                  && (Browser_Version_Haupt >= 4)
                  && (Browser_Version_Haupt < 5)
                );

var NS6x  =    NSab6;

                // theroretisch kann noch auf die Hauptversionsnummer geprüft werden
                //      var NS6x = (      (NSab6)
                //                      && (Browser_Version_Haupt >= 6)
                //                      );

var IE    =    (document.all) ? true : false;    // if-Anweisung ist wichtig, sonst erfolgt Zeigerzuweisung

var IEab5 =    (      (IE)
                  && (document.getElementById )    // ab IE 5.x ist getElementById
                  implementiert
                );

var IE4x  =    (      (IE)                // NS-Browser setzt IE    auf false
                  && (!IEab5 )            // NS-Browser setzt !IEab5 auf true
                                          // Es muss (IE) && (!IEab5) abgefragt
                                          //      Würde nur (!IEab5) kodiert
                                          //      dann würde der NS-Browser
                                          //      IE4x auf true setzen
                  && (Browser_Version_Haupt >= 4)
                  && (Browser_Version_Haupt < 5)
                );

var IE5x  =    (      (IEab5 )
                  && (Browser_Version_Haupt >= 5)
                );

```




```

        && (Browser_Version_Haupt < 6)
    );

    var IE55 = (
        (IEab5)
        && (Browser_Version_Haupt == 5)
        && (Browser_Version_Unter == 5)
    );

    var IE6x = (
        (IEab5)
        && (Browser_Version_Haupt >= 5)
    );

```

Wiedergabe von Medien mit dem Windows Media Player und HTML-Navigation im Dokument:

Die aktive Wiedergabe eines Mediums wird mit Wechsel des Dokumentes z.B. per <A HREF....> automatisch gestoppt. Der Player ist also nur im Dokument verwendbar, in das er eingebettet ist und das nur solange, wie das Dokument geladen ist.

Vergleich .style.time2 Behavior und Windows Media Player ab 7.1:

Die Objekte und Collectionen zum Player und zum Behavior .style.time2 sind konzeptionell ähnlich:

Der Player und .style.time2 basieren auf gemeinsamen Prinzipien, nur dass sich die referenzierten Objekte und Collectionen anders nennen bzw. verschieden aufgebaut sind.

Die Eigenschaften zum .style.time2 Behavior und zum Windows Media Player sind sehr ähnlich, wenn nicht gar identisch.

Während der Windows Media Player alle Windows-kompatiblen Medienarten unter einem gemeinsamen Dach verwalten kann, muss beim .style.time2 Behavior das richtige Behavior-Objekt gewählt werden, um entsprechende spezielle Methoden zum Typ des Mediums ansprechen zu können.

Das Objektmodell des Windows Media Players ist wesentlich ganzheitlicher, dadurch übersichtlicher, und hat einen größeren Umfang als das Behaviors. Z.B. kann nur der Player CD-Laufwerke als Systemkomponente verwalten.

Die Programmierung von .style.time2 ist durch die Verwendung des Players vielleicht unnötig, setzt aber voraus, dass der Player auch verwendet werden **soll**.

Per .style.time2 Behavior kann der Windows Media Player eingebunden werden, muss aber nicht. Wird der Player eingebunden, so ist er per Behavior ansprechbar. Die Programmierung des Players setzt **aber** eine permanente Player-Instanz im HTML-Dokument voraus und damit ein permanentes Playerfenster. Außerdem wird der Player im **HTML-Dokument** nicht mit allen seine Objekteigenschaften unterstützt, was vielleicht doch wieder für die Nutzung des Behavior .style.time2 spricht.

Es ist zu vermuten, dass

der Behavior ebenfalls das Active-X-Control benutzt, wenn der Windows Media Player in den Behavior eingebunden ist.

die Verwendung des Behaviors ressourcenschonender ist, denn es werden nur Eigenschaften und Methoden instanziiert, die für das Behavior-Objekt nötig sind.

Behavior .style.time2 wird per XML-Tag in das HTML-Dokument eingebunden. Der Player wird in das HTML-Dokument per Active-X-Control im OBJECT-Tag eingebunden.

Beispiele zum Windows Media Player:

Alle Beispiele dienen nur zum Verständnis der Programmierung unter JScript.

5.2.4.1. Begriffe

5.2.4.1.1. Media Datei

Eine Media Datei

kann lokal auf dem User-PC oder im Netzwerk (z.B. Internet) liegen

ist Video und Audio in diversen Formen z.B.:

nicht window-spezifisch:	*.AVI
	*.MID
	*.MP3
	*.MPEG
	*.WAV
window-spezifisch	*.WM
	*.WMA
	*.WMV
	*.ASF

Datei-Suffix	MIME type	Windows-Media-Datei
*.wma	audio/x-ms-wma	nur Audio
*.wmv	video/x-ms-wmv	Audio/Video
*.asf	video/x-ms-asf	Audio/Video

hat Media-Typ z.B. "Audio" oder "Video"



kann Attribute folgende Attribute besitzen

"Album"	nur bei Media Item
"Artist"	nicht für Playlist, die per Methode ID_Player.mediaCollection.getByXXX() oder ID_Player.mediaCollection.getAll() erzeugt wurde
"Author"	
"Bitrate"	Bitrate, nur bei Element aus Media-Bibliothek
"Copyright"	nur bei Playlist-Eintrag nicht Playlist von CD nicht Media-Item
"CreationDate"	Datum des Hinzufügen des Elementes zur Media-Bibliothek nicht Playlist und Media Item
"DigitallySecure"	Datum des Schutzes des Elementes in der Media-Bibliothek Schutz = Digital Rights Management nicht Playlist und Media Item
"Genre"	nur Playlist-Eintrag nicht Playlist von CD
"MediaType"	Media-Typ (audio oder video)
"Name"	Name des Playlist-Eintrages
"PlayCount"	Anzahl der Wiederholungen eines Elementes aus der Media-Bibliothek nicht Playlist und Media Item
"SourceURL"	Url oder Pfad und Dateiname der Media-Datei als Element in der Media Bibliothek nicht Playlist und Media Item
"TOC"	CD Table of Contents Identifier nur für Playlist von CD nicht für Playlist, die per Methode ID_Player.mediaCollection.getByXXX() oder ID_Player.mediaCollection.getAll() erzeugt wurde

Die Attribute haben natürlich je nach Media-Datei einen konkreten Inhalt.

5.2.4.1.1.1. **Media Datei als nicht window-spezifische Media Datei**

Die Media Datei enthält **nur** Daten des Medium, das nicht nur unter Windows verwaltet werden kann.

z.B. Sound wie MP3 etc.
Media Clip als Video mit Sound.

Diese Art von Media Dateien existiert z.T. schon seit DOS-Zeiten, also vor Windows.

5.2.4.1.1.2. **Media Datei als window-spezifische Media Datei**

5.2.4.1.1.2.1. **Media Datei als Windows Media Datei**

Die Media Datei enthält **nur** Daten des Medium, das nur unter Windows verwaltet werden kann.

Typisches Kennzeichen (leider nicht immer) dieser Media Datei ist der Suffix *.Wxx mit **W** für Windows

z.B. Dateien mit Suffix *.wma
*.wmv
aber auch *.asf

5.2.4.1.1.2.2. **Media Datei als Windows Meta Datei**

Eine Meta Datei ist eine Script-Datei (z.B. ASX-Datei), die

spezifische Meta-Elemente zur Verwaltung von Media Dateien und Media Daten besitzt
folgende Media Dateien laden kann:

nicht-window-spezifische	Media Datei(en) wie MP3-Sound oder Videoclip
window-spezifische	Media Datei(en), aber nur dann, wenn die Windows-Kompressionstechnologie zur Medienart auch in der Verwaltung implementiert ist
window-spezifische	Meta Datei(en), aber nur teilweise

Übersicht zu Windows Meta Dateien und ladbaren Windows Media Dateien:

Meta-Datei-Suffix	Windows-Media-Datei-Suffixe
	*.asf *.wma *.wmv



*.wvx	X	X	X
*.wax	X	X	
*.asx	X		

X = ladbar

Übersicht zu Windows Meta Dateien:

Mime-Typ der	Windows Meta	Datei
ladbare	Windows Media	Datei
ladbare	Windows Meta	Datei

Meta Datei-Suffix	MIME Type	Windows Media-Datei-Suffix	bzw. Meta-Datei-Suffix
*.wax	audio/x-ms-wax	Media-Datei mit Suffix Meta-Datei mit Suffix	*.asf *.wma *.wax
*.wvx	video/x-ms-wvx	Media-Datei mit Suffix Meta-Datei mit Suffix	*.wma *.wmv *.wax *.wvx
*.asx	video/x-ms-asf	Media-Datei mit Suffix Meta-Datei mit Suffix	*.asf *.wma *.wmv *.asx *.wax *.wvx

5.2.4.1.2. Media Bibliothek

Die Media-Bibliothek ist eine Sammlung

von media Objekten: auf Basis der Collection mediaCollection

von Playlisten: Playlist besteht aus mindestens 2 Media Items als gelistete media Objekte
Jede Playliste hat einen eindeutigen Namen.

als Datenbank: Namensindex (Namen der Objekte)
Datenbank wird vom Player automatisch verwaltet und gespeichert.

als Datei-Pool: als Menge physischer Verknüpfungen bei lokalen Media Dateien auf der Festplatte
als Menge logischer Verknüpfungen bei nicht-lokalen Media Dateien (z.B. im Netz)

5.2.4.1.3. Playliste

Eine Playliste ist Liste von mindestens 2 media Objekten aus der Media-Bibliothek.

In der Playliste wird ein media Objekt als Media Item Objekt bezeichnet

Eine Playliste kann

lokal oder im Netzwerk liegen
Datenbank oder Textdatei sein

5.2.4.1.4. JScript: media Objekt, Media Item Objekt und Collection mediaCollection

Ein media Objekt ist

eine media-typ-gerecht instanzierte Media-Datei mit den Attributen der Media-Datei
ein Media Item Objekt als Playlisten-Eintrag, der einer Playliste zugeordnet ist.

Alle media Objekte werden in der Collection mediaCollection verwaltet (jedes Element ist ein media Objekt)

Das **aktuelle** media Objekt wird gerade im Player wiedergegeben.

Hinweis: Es kann zu jedem Zeitpunkt nur 1 Objekt im Player wiedergegeben werden, also aktuell sein.

5.2.4.2. Varianten des Windows Media Player (der ActiveX-Controls)

Variante	Wert des CLASSID-Attributes im OBJECT-Tag
Windows Media Player in minimaler Version der Control-Elemente	05589FA1-C356-11CE-BF01-00AA0055595A
Windows Media Player in erweiterter Version der Control-Elemente	22d6f312-b0f6-11d0-94ab-0080c74c7e95
Media Bar Player, also Windows Media Player	52ca3bcf-3b9b-419e-a3d6-5d28c0b0b50c
Achtung: CLASSID wird inzwischen nicht mehr sicher unterstützt !	
Windows Media Player 7.1	6BF52A52-394A-11D3-B153-00C04F79FAA6

Hinweis: Groß-Kleinschreibung des Wertes vom CLASSID-Attributes ist egal.

Beispiele für Active-X-Control des Mediaplayers zum Abspielen eines Videos oder Sounds:

Achtung: CLASSID wird inzwischen nicht mehr sicher unterstützt (siehe unten Microsoft ändert die Active-X-Eigenschaften)

```
<BODY>
<OBJECT ID="ActiveMovie1"
        WIDTH=356
        HEIGHT=328
        CLASSID="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A"
>
    <PARAM NAME="MovieWindowWidth" VALUE="352">
    <PARAM NAME="MovieWindowHeight" VALUE="247">
    <PARAM NAME="FileName" VALUE="0.wmv">
</OBJECT>
```

Das Active-X-Control umfasst folgende PARAM



```

<PARAM NAME="Appearance" VALUE="0">
<PARAM NAME="AutoStart" VALUE="0">
<PARAM NAME="AllowChangeDisplayMode" VALUE="-1">
<PARAM NAME="AllowHideDisplay" VALUE="0">
<PARAM NAME="AllowHideControls" VALUE="-1">
<PARAM NAME="AutoRewind" VALUE="-1">
<PARAM NAME="Balance" VALUE="0">
<PARAM NAME="CurrentPosition" VALUE="0">
<PARAM NAME="DisplayBackColor" VALUE="0">
<PARAM NAME="DisplayForeColor" VALUE="16777215">
<PARAM NAME="DisplayMode" VALUE="0">
<PARAM NAME="Enabled" VALUE="-1">
<PARAM NAME="EnableContextMenu" VALUE="-1">
<PARAM NAME="EnablePositionControls" VALUE="-1">
<PARAM NAME="EnableSelectionControls" VALUE="0">
<PARAM NAME="EnableTracker" VALUE="-1">
<PARAM NAME="Filename" VALUE="">
<PARAM NAME="FullScreenMode" VALUE="0">
<PARAM NAME="MovieWindowSize" VALUE="0">
<PARAM NAME="PlayCount" VALUE="1">
<PARAM NAME="Rate" VALUE="1">
<PARAM NAME="SelectionStart" VALUE="-1">
<PARAM NAME="SelectionEnd" VALUE="-1">
<PARAM NAME="ShowControls" VALUE="-1">
<PARAM NAME="ShowDisplay" VALUE="-1">
<PARAM NAME="ShowPositionControls" VALUE="0">
<PARAM NAME="ShowTracker" VALUE="-1">
<PARAM NAME="Volume" VALUE="-600">

```

VALUE-Angaben sind nur Beispiele

Bezeichner der Eigenschaften für JScript sind IDENTISCH mit den Werten aus NAME-Attribut von PARAM !
Gross-Kleinschreibung beachten !

User	AllowChangeDisplayMode	Änderung des Anzeigemodus Zeitanzeige bzw. Framesanzeige durch User boolean true oder false
	AllowHideControls	Änderung des Anzeigemodus der Steuerungstasten während Wiedergabe durch User boolean true oder false false (0) für nicht erlauben
	AllowHideDisplay	Änderung des Anzeigemodus der Wiedergabe-Anzeige während Wiedergabe durch boolean true oder false false (0) für nicht erlauben
	AutoRewind	Automatisches Rückspulen nach Stop boolean true oder false false (0) für nicht rückspulen
	AutoStart	Autostart der Wiedergabe ein oder aus boolean true oder false false (0) für nicht automatische Wiedergabe: siehe Aktionen zum Player
	Balance	numerisch 0 für zentriert default -1000 ganz links +1000 ganz rechts
	CurrentPosition	unklar
	DisplayBackColor	Hintergrundfarbe der Control-Elemente numerischer Farbwert, kein Hexastring 0 Default für Black 16777215 für Weiss
	DisplayForeColor	Hintergrundfarbe der Control-Elemente numerischer Farbwert, kein Hexastring 0 Default für Black 16777215 für Weiss
	DisplayMode	Anzeigeart der Position 0 für Time



	1 für Frames
Enabled	Control-Anzeige ein, aus boolean true oder false false (0) für nicht erlauben
EnableContextMenu	Rechte-Maus-Kontextmenü ein, aus boolean true oder false false (0) für nicht erlauben
EnablePositionControls	Positionierungs-Button im Control-Panel ein, aus boolean true oder false false (0) für nicht erlauben
EnableSelectionControls	Selektions-Button im Control-Panel ein, aus boolean true oder false false (0) für nicht erlauben
EnableTracker	Tracker-Balken im Control-Panel ein, aus boolean true oder false false (0) für nicht erlauben
FileName	Mediendatei mit Pfad oder Url
FullScreenMode	Vollbildschirm ein, aus boolean true oder false false (0) für kein Vollbildmodus
MovieWindowSize	unklar, Grösse des Anzeigefensters
PlayCount	Anzahl der Wiedergaben der Mediendatei ab 1
Rate	Wiedergabefaktor numerisch 1 für 1-fache Geschwindigkeit 10 für 20-fache Geschwindigkeit
SelectionEnd	Bis-Position, bis zu der wiedergegeben werden soll in Sekunden bezüglich Anfang der Mediendatei (0)
SelectionStart	Von-Position, bis zu der wiedergegeben werden soll in Sekunden bezüglich Anfang der Mediendatei (0)
ShowControls	Start Stop buttons and tracker controls sichtbar ein, aus boolean true oder false false (0) für aus
ShowDisplay	Status-Anzeige-Panel sichtbar ein, aus boolean true oder false false (0) für aus
ShowPositionControls	Previous, Rewind, Forward and Next buttons sichtbar ein, aus boolean true oder false false (0) für aus
ShowSelectionControls	Selektions-Button im Control-Panel sichtbar ein, aus boolean true oder false false (0) für aus
ShowTracker	Trackerbar im Controlpanel sichtbar ein, aus: nur lesen boolean true oder false false (0) für aus
uiMode	Sichtbarkeit anstelle STYLE-visible "invisible" für unsichtbar
Volume	Laustärke als Basis dient die aktuelle Windows-Lautstärke integer 0 für maximimale, also aktuelle Windows-Lautstärke (höher geht nicht !!!) -10000 für total still

Den ganzen Mediaplayer unsichtbar machen: per STYLE-Attribut im OBJECT-Tag



document.ID-Bezeichner.readyState (Mediaplayer 6.4, funktioniert aber auch mit höheren Versionen)

.readyState hat folgende Werte

0	The FileName property has not been initialized.
1	Windows Media Player control is asynchronously loading a file.
3	Windows Media Player control loaded a file, and downloaded enough data to play the file, but has not yet received all data.
4	All data has been downloaded.

Methoden:

.IsSoundCardEnabled()

liefert boolean
true Soundkarte erkannt
false keine Soundkarte erkannt

.Run(); Erzeugt und startet Instanz des Mediaplayers und ermöglicht danach weitere Aktionen wie .Play etc.

.Mute ohne ()

lesen, schreiben
true Player volume is muted.
false (Default) Player volume is not muted.

.Play ohne ()

Wiedergabe starten

.AutoStart ohne ()

true (Default) Clip automatically starts.
false Disables automatic start.

.Stop ohne ()

.Pause ohne ()

Audio wiedergeben

```
<HEAD>
<SCRIPT>
var SoundStarten_TimeoutID=0;

function SoundStarten()
{
  // auf Soundkarte prüfen
  if (ID_PlayerObjekt.IsSoundCardEnabled())
  {
    // Soundkarte vorhanden
    // Musik geladen und abspielfähig ?
    if (ID_PlayerObjekt.readyState == 4)
    {
      // geladen, also Playerinstanz starten
      ID_PlayerObjekt.Run();
      // Musik wiedergeben
      ID_PlayerObjekt.Play();
      alert('Musik ist gestartet !');
    }
    else{ SoundStarten_TimeoutID=window.setTimeout('SoundStarten()', 100);}
  }
  else{ alert('Keine Soundkarte gefunden !');}
}
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_PlayerObjekt" CLASSID="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A">
  <PARAM NAME="FileName" VALUE="a0.mp3">
  <PARAM NAME="AutoStart" VALUE="0">      <!-- kein Autoplay -->
</OBJECT>
<SCRIPT>
if (ID_PlayerObjekt != null){SoundStarten();}
</SCRIPT>
</BODY>
```



</HTML>

5.2.4.3. Instanziierung des Windows Media Player im HTML-Dokument

Zunächst erst ein Beispiel, das zur **Verdeutlichung** wichtiger Kriterien der Instanziierung dient:

```
<HTML>
<HEAD>
<SCRIPT>
<!--
    function WiedergabeStarten()
    { ID_Player.controls.play(); }

    function WiedergabeStoppen()
    { ID_Player.controls.stop(); }

-->
</SCRIPT>
</HEAD>
<BODY>
    <OBJECT          ID="ID_Player"
                    CLASSID="CLSID:6BF52A52-394A-11d3-B153-00C04F79FAA6"
    >
        <PARAM NAME="URL" VALUE="test.wma">
    </OBJECT>
    <INPUT TYPE="BUTTON" NAME="StartButton" VALUE="Play" onclick="WiedergabeStarten()">
    <INPUT TYPE="BUTTON" NAME="StopButton" VALUE="Stop" onclick="WiedergabeStoppen()">
</BODY>
</HTML>
```

5.2.4.3.1 Belegung CLASSID-Attribut im OBJECT-Tag

Variante	Wert des CLASSID-Attributes im OBJECT-Tag
Windows Media Player in minimaler Version der Control-Elemente	05589FA1-C356-11CE-BF01-00AA0055595A
Windows Media Player in erweiterter Version der Control-Elemente	22d6f312-b0f6-11d0-94ab-0080c74c7e95
Media Bar Player, also Windows Media Player	52ca3bcf-3b9b-419e-a3d6-5d28c0b0b50c
Windows Media Player 7.1	6BF52A52-394A-11D3-B153-00C04F79FAA6

Hinweis: Groß-Kleinschreibung des Wertes vom CLASSID-Attributes ist egal.

5.2.4.3.2. ID-Attribut für die Referenz auf die Instanz des Windows Media Players

Der Wert des ID-Attributes im OBJECT-Tag wird als Referenz auf das Objekt des Windows Media Player benutzt. Über dieses ID kann auf sämtliche Objekte und Collectionen zum Windows Media Player zugegriffen werden.

In der nachfolgenden Beschreibung zum Windows Media Player wird "ID_Player" synonym zum Begriff "Objekt des Windows Media Players" verwendet.

5.2.4.3.3. HTML-Vorbelegung von Eigenschaften der Instanz des Windows Media Players

Nicht alle Eigenschaften des Windows Media Player sind schreibbar, also mit Wert belegbar. HTML bietet nur einen Ausschnitt der implementierten Eigenschaften des Windows Media Players.

Das PARAM-Tag innerhalb des OBJECT-Tags lässt die Vorbelegung einer **schreibbaren** Eigenschaft des Windows Media Player per HTML zu. Die Vorbelegung ist optional und kann auch komplett in JScript erfolgen (dort mit allen implementierten und schreibbaren Eigenschaften).

Pro Eigenschaft, die in HTML belegbar ist, existiert ein PARAM-Tag:

Name der Eigenschaft laut NAME-Attribut

Wert der Eigenschaft laut VALUE-Attribut

Objekt	Eigenschaft als String-Wert in NAME
ID_Player	enabled enableContextMenu fullScreen stretchToFit uiMode URL
ID_Player.closedCaption	captioningID SAMIFilename SAMILang SAMISyle
ID_Player.controls	currentMarker currentPosition
ID_Player.currentMedia	duration name



ID_Player.settings	autoStart
	balance
	baseURL
	defaultFrame
	enableErrorDialogs
	invokeURLs
	mute
	playCount
	rate
	volume

Hinweis: Groß-Kleinschreibung des Wertes im NAME-Attribut ist egal.

Der Wert des VALUE-Attributes im PARAM-Tag ist der Wert der Eigenschaft, mit dem sie vorbelegt wird.

Mögliche Werte des VALUE-Attributes: siehe entsprechende Objektbeschreibung

Beispiele zum Sinn der Vorbelegung von Eigenschaften:

Control-Elemente des Players:

Wenn Control-Elemente des Players sichtbar sind, dann kann der User die Wiedergabe steuern. Die Art der Control-Elemente wird automatisch zum Typ der Media-Datei erzeugt (Tasten zur Steuerung der

Wiedergabe

durch den User, wobei diese zum Typ der Media-Datei passen).

Sollen die Control-Elemente generell sichtbar oder nicht sichtbar sein, so ist das auch per Script über die Eigenschaft ID_Player.enabled einstellbar.

Soll der Umfang der Control-Elemente eingestellt werden, so ist das auch per Script über die Eigenschaft ID_Player.uiMode möglich.

Videogröße:

Es kann sein, dass die widerzugebende Media-Datei einen größeren Platz braucht, als das aktuelle Player-Fenster. Dann muss der User das Fenster manuell vergrößern. Das lässt sich aber vermeiden.

Alternativ kann auch per Script die Eigenschaft ID_Player.stretchToFit verwendet werden.

5.2.4.3.4. HTML-Attribute WIDTH und HEIGHT im OBJECT-Tag

Diese Attribute bestimmen die Dimension des Playerfenster und die Möglichkeit der Darstellung von Control-Elementen (Tasten zur Steuerung der Wiedergabe durch den User, wobei diese zum Typ der Media-Datei passen).

Eine Höhenangabe muss mindestens 41 Pixel sein, damit überhaupt Control-Elemente des Players sichtbar werden können. 41 Pixel benötigt die Leiste der Control-Elemente mindestens.

Alternativ zu Angaben per PARAM-Attribute im OBJECT-Tag kann die Anzeige von Control-Elementen auch per Script gesteuert werden:

Sollen die Control-Elemente generell sichtbar oder nicht sichtbar sein, so ist das per Script über die Eigenschaft

ID_Player.enabled

einstellbar.

Soll der Umfang der Control-Elemente eingestellt werden, so ist das per Script über die Eigenschaft

ID_Player.uiMode

einstellbar.

Es kann sein, dass die widerzugebende Media-Datei einen größeren Platz braucht, als das aktuelle Player-Fenster. Dann muss der User das Fenster manuell vergrößern. Alternativ kann auch per Script die Eigenschaft ID_Player.stretchToFit verwendet werden.

5.2.4.3.5. automatische Fehleranzeige

Die automatische Fehleranzeige ist ein- bzw. abschaltbar per Eigenschaft ID_Player.settings.enableErrorDialogs

5.2.4.3.6. Beispiel

Hinweis: In diesem Beispiel wurden Kommentare innerhalb von HTML ungeachtet der Syntaxvorschriften eingearbeitet.

Im Beispiel werden GIF-Dateien verwendet:

Wenn im Datei-Namen die Zeichenkette "_hi" auftaucht, dann ist es die **aufgehellte** Grafik der Normalversion

Wenn im Datei-Namen die Zeichenkette "_low" auftaucht, dann ist es die **abgedunkelte** Grafik der Normalversion

Bsp.:	stop.gif	Normalversion
	stop_hi.gif	aufgehellte Normalversion
	stop_low.gif	abgedunkelte Normalversion

<HTML>

<HEAD>

<SCRIPT>

```
var ActiveXAktiv; // ist true wenn ActiveX benutzt wird
                  // ist sonst false
```

```
var timerID;
```




```

function Init()      // mit Start des Dokumentes aktiviert im BODY
{
    ActiveXAktiv=false; // Annahme: kein ActiveX

    // prüfen auf ActiveX
    if ( (navigator.userAgent.indexOf('IE') > -1)
        && (navigator.platform == "Win32")
        )
    {
        // ActiveX erkannt
        ActiveXAktiv = true;
    }

    // Player starten anstelle Autostart im OBJECT-Tag
    Starten();
}

// Eventhandler
function EventHandler(NewState)
{
    // alle Bilder in abgedunkelter Version anzeigen
    timerID = setTimeout("document.all.Name_Image1.src = 'play_low.gif';",0);
    timerID = setTimeout("document.all.Name_Image2.src = 'pause_low.gif';",0);
    timerID = setTimeout("document.all.Name_Image3.src = 'stop_low.gif';",0);

    // den neuen Status abklappern und Bilder in dementsprechend anzeigen
    switch(NewState)
    {
        case 1:
            timerID = setTimeout("document.all.Name_Image3.src = 'stop_hi.gif';",0);
            alert('gestoppt');
            break;
        case 2:
            timerID = setTimeout("document.all.Name_Image2.src = 'pause_hi.gif';",0);
            alert('pausiert');
            break;
        case 3:
            timerID = setTimeout("document.all.Name_Image1.src = 'play_hi.gif';",0);
            alert('gestartet');
            break;
    }
}

function BildWechsel(ID_BildAlsKette,whichImage, URL_BildAlsKette)
{
    // Wechsel nur, wenn es kein aufgehelltes Bild ist
    if ( ID_BildAlsKette.src.indexOf('_hi') == -1 )
    { ID_BildAlsKette.src = URL_BildAlsKette;}
}

function Starten()
{
    if (ActiveXAktiv == true)
    {
        // ActiveX
        document.all.ID_Player.controls.play();
    }
}

function Stoppen()
{
    if (ActiveXAktiv == true)
    {
        // ActiveX

        // Wiedergabe stoppen
        document.all.ID_Player.controls.stop();

        // Player auf Anfang setzen, also aktuelle Position in Media-Datei ist 0
        document.all.ID_Player.controls.currentPosition=0;
    }
}

function Pausieren() // Status muss 3 sein also pausieren

```



```

    {
        if (ActiveXAktiv == true)
        {
            // ActiveX
            if (document.all.ID_Player.playState == 3)
            { document.all.ID_Player.controls.pause(); }
        }
    }
</SCRIPT>
</HEAD>

<BODY onload="Init()">
    <SCRIPT FOR="ID_Player" EVENT="playStateChange(NewState)" LANGUAGE="JScript">
        // IE-Event-Capturing starten
        EventHandler(NewState);
    </SCRIPT>

    <OBJECT ID="ID_Player"
        CLASSID="CLSID:6BF52A52-394A-11D3-B153-00C04F79FAA6"
        WITH=176
        HEIGHT=144
    >
        <PARAM NAME="AutoStart" VALUE="False">
        <PARAM NAME="URL" VALUE="test.wma">
    </OBJECT>

    <TABLE>
    <TR>
    <TD>
        <A HREF="#" onmouseover="BildWechsel(
        Name_Image1, 'play.gif')
        onmouseout="BildWechsel(
        Name_Image1, 'play_low.gif')
        onclick="Starten()"
        >
            <IMG NAME="Name_Image1" SRC="play_low.gif" BORDER=0>
        </A>
    </TD>
    <TD>
        <A HREF="#" onmouseover="BildWechsel(
        Name_Image2, 'pause.gif')
        onmouseout="BildWechsel(
        Name_Image2, 'pause_low.gif')
        onclick="Pausieren()"
        >
            <IMG NAME="Name_Image2" SRC="pause_low.gif" BORDER=0>
        </A>
    </TD>
    <TD>
        <A HREF="#" onmouseover="BildWechsel(
        Name_Image3, 'stop.gif')
        onmouseout="BildWechsel(
        Name_Image3, 'stop_low.gif')
        onclick="Stoppen()"
        >
            <IMG NAME="Name_image3" SRC="stop_low.gif" BORDER=0>
        </A>
    </TD>
    </TR>
    </TABLE>
</BODY>
</HTML>

```

5.2.4.4. Eventbehandlung

Events laufen beim Windows Media Player nur auf der obersten Ebene, also auf der Ebene der Instanz laut ID-Attribut im OBJECT-Tag, ein.

Zunächst erst ein Beispiel, das zur Verdeutlichung wichtiger Kriterien der Eventbehandlung dient:

```

<HEAD>
<SCRIPT>
    function EventHandler(NewState)
    {
        // hier die Massnahmen kodieren
        // es kann der Parameter NewState weiterverarbeitet werden, muss aber nicht
    }
</SCRIPT>
</HEAD>
<BODY>

```



```

<SCRIPT FOR="ID_Player" EVENT="playStateChange(NewState)" LANGUAGE="JScript">
    EventHandler(NewState);
</SCRIPT>
<BODY>

```

Die im HEAD kodierte EventHandler-Routine

hat einen freien Funktionsbezeichner (im Beispiel den Bezeichner EventHandler)
muss Parameter **exakt** in der Schreibweise, die zum genau Event definiert ist, besitzen
(im Beispiel **NewState** (und nicht newstate etc.))

Das <SCRIPT FOR></SCRIPT> aktiviert das Abfangen des Ereignisses und bindet den Eventhandler ein, der das Ereignis verwaltet.

Es ist übersichtlicher, den Code des Eventhandlers in eine Funktion zu legen. Eine direkte Script-Kodierung zwischen
<SCRIPT FOR ...></SCRIPT> ist jedoch auch möglich.

Scriptsprache kann nur "JScript" sein !

Der gesamte Scriptcode des HTML-Dokumentes darf die vordefinierten Eventbezeichner und Event-Parameter nicht neu definieren, da sonst die Eventbehandlung scheitert.

Bsp.: Der Bezeichner des Eventhandlers darf nicht identisch sein mit dem Bezeichner eines Events.
Der Bezeichner des Events darf nicht re-definiert werden.

Eventbezeichner und Event-Parameter haben **globale Gültigkeit** im gesamten Dokument, das den Windows Media Player instanziiert hat.

5.2.4.5. Objekte, Collectionen und Events des Windows Media Player

Der Wert des ID-Attributes im OBJECT-Tag wird als Referenz auf das Objekt des Windows Media Player benutzt. Über dieses ID kann auf sämtliche Objekte und Collectionen zum Windows Media Player zugegriffen werden.

In der nachfolgenden Beschreibung zum Windows Media Player wird "ID_Player" synonym zum Begriff "Objekt des Windows Media Players" verwendet.

| | | |
|------------|------------------------|--|
| Beispiele: | aktuelles media Objekt | ID_Player.currentMedia |
| | aktuelles Media Item | ID_Player.controls.currentItem |
| | aktuelle Playliste | ID_Player.currentPlaylist |
| | beliebige Playliste | ID_Player.playlistCollection.item() |

5.2.4.5.1. Objekt des Windows Media Players (ID_Player)

Über ID_Player, also den Zeiger auf die Instanz, also auf das Objekt des Windows Media Players im HTML-Dokument, lassen sich grundlegende Eigenschaften und Methoden verwalten
Events auslösen und behandeln: Die Events aller Objekte und Collectionen landen auf der Ebenen der Instanz. laut ID_Player.

Die Programmierung von Skins (Layouts zum Player) wird nicht beschrieben.

Syntax:

```

ID_Player.eigenschaft
ID_Player.methode()

```

Eigenschaften:

Nicht für Skins verwendbar sind folgende Eigenschaften:

```

ID_Player.enableContextMenu
ID_Player.enabled
ID_Player.fullScreen
ID_Player.uiMode

```

| | |
|--------------------|--|
| .cdromCollection | Zeiger auf Collection cdromCollection |
| .closedCaption | Zeiger auf Objekt closedCaption |
| .controls | Zeiger auf Objekt controls |
| | Hinweis: aktuelles media Objekt ID_Player.currentMedia |
| | aktuelles Media Item ID_Player.controls.currentItem |
| | aktuelle Playliste ID_Player.currentPlaylist |
| | Playliste ID_Player.playlistCollection.item() |
| .currentMarker | Nummer des aktuellen Marker in der Media-Datei |
| | Marker als aktuellen Marker setzen |
| | Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount |
| | Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei |
| .currentPlaylist | Zeiger auf Objekt currentPlaylist |
| | Hinweis: aktuelles media Objekt ID_Player.currentMedia |
| | aktuelles Media Item ID_Player.controls.currentItem |
| | aktuelle Playliste ID_Player.currentPlaylist |
| | Playliste ID_Player.playlistCollection.item() |
| .enableContextMenu | Kontextmenü ein bzw. aus bei rechte Maus-Click |
| .enabled | Windows Media Player-Control-Elemente ein bzw. aus (true bzw. false) |
| .error | Zeiger auf Objekt error |
| .fullScreen | Vollbildmodus bei Wiedergabe eines Video ein bzw. aus (true bzw. false) |
| .isOnline | prüfen ob Client on- bzw. offline zum Netzwerk ist (z..B Internet) (true bzw. false) |



| | |
|---------------------|--|
| .mediaCollection | <p>Zeiger auf Collection mediaCollection</p> <p>Hinweis: aktuelles media Objekt ID_Player.currentMedia
 aktuelles Media Item ID_Player.controls.currentItem
 aktuelle Playliste ID_Player.currentPlaylist
 Playliste ID_Player.playlistCollection.item()</p> |
| .network | Zeiger auf Objekt network |
| .openState | aktueller Status des Players bezüglich Playliste, Media-Datei, Codec, Lizenz, Individualisierung |
| .playlistCollection | <p>Zeiger auf Collection playlistCollection</p> <p>Hinweis: aktuelles media Objekt ID_Player.currentMedia
 aktuelles Media Item ID_Player.controls.currentItem
 aktuelle Playliste ID_Player.currentPlaylist
 Playliste ID_Player.playlistCollection.item()</p> |
| .playState | <p>Status der Wiedergabe der aktuellen Media-Datei (Status des Players bei Wiedergabe)</p> <p>Media-Datei muss bestimmte Wiedergabe-Möglichkeiten unterstützen, damit deren Status angezeigt wird</p> <p>Download und Wiedergabe parallel möglich bei Dateien mit Suffix</p> <p>*.ASF
 *.AVI
 *.MP3
 *.MPEG
 *.WAV
 *.WM
 *.WMA
 *.WMV</p> |
| | <p>Integer</p> <p>0 undefined
 1 Wiedergabe ist gestoppt
 2 Wiedergabe pausiert
 3 Wiedergabe erfolgt fortlaufend
 4 Wiedergabe mit schnellem Vorlauf
 5 Wiedergabe mit schnellem Rücklauf
 8 Wiedergabe ist komplett und geendet
 10 Wiedergabe kann beginnen, da Player bereit dafür ist</p> |
| .settings | Zeiger auf Objekt settings |
| .status | interner und laufender Status des Players |
| .stretchToFit | <p>Playerfenster bei Media-Datei als Video in der Ansicht automatisch auf Videogröße ausdehnen ein bzw. aus</p> <p>sinnvoll, wenn Video größere Dimension haben könnte als das Layout des Players (Player-Fenster)</p> <p>true, so Playerfenster automatisch anpassen auf Videogröße</p> <p>false Default
 Playerfenster nicht automatisch anpassen auf Videogröße</p> |
| .uiMode | <p>Umfang der anzuzeigenden Control-Elemente des Players aus der Menge aller Control-Elemente zum aktuellen Media-Datei-Typ</p> <p>Control-Elemente kann der User zur Steuerung der Wiedergabe benutzen</p> <p>Control-Elemente nur sichtbar, wenn der Player sichtbar ist</p> <p>falls im OBJECT-Tag das HEIGHT-Attribut kodiert wurde, dann mit Wert > 40 Pixel</p> <p>String</p> <p>"none" keine Control-Elemente anzeigen
 User kann nichts steuern
 Achtung: Im OBJECT-Tag stets HEIGHT=0 und WIDTH=0 setzen</p> <p>"mini" angezeigt werden nur die Elemente
 Pause-Taste
 Stop-Taste
 Mute-Taste
 Lautstärkeregelung</p> <p>"mini" angezeigt werden alle Elemente</p> |
| .URL | <p>Url der Media-Datei, die wiedergegeben werden soll</p> <p>Media-Datei wird geladen und zum aktuellen Medium (aktuelles media Objekt)</p> <p>Autostart der Wiedergabe: laut ID_Player.settings.autoStart</p> <p>wenn Autostart erlaubt:
 Wiedergabe sofort gestartet bei Zuweisung von Werten zu
 ID_Player.URL
 ID_Player.currentMedia</p> <p>wenn Autostart nicht erlaubt ist:
 Wiedergabe nach Zuweisung von Werten zu
 ID_Player.URL
 ID_Player.currentMedia</p> <p>durch Aufruf von ID_Player.controls.play()</p> <p>Hinweis: Zuweisung zu
 ID_Player.URL
 ID_Player.currentMedia</p> |



| | |
|--|---|
| | setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht |
| .versionInfo | siehe ID_Player.currentMedia |
| Methoden: | Version der instanziierten Windows Media Player |
| .close() | schliesst den Player im HTML-Dokument |
| .launchURL() | eine Media-Datei an den Standardbrowser im System schicken |
| Events: | |
| automatische Fehleranzeige ein- bzw. abschaltbar per | ID_Player.settings.enableErrorDialogs |
| buffering | erzeugt, wenn das Puffern von Media-Daten startet bzw. endet |
| Syntax: | buffering(Start) |
| | Start true, so Puffern starten |
| | false, so Puffern stoppen |
| | Hinweis: Download und Wiedergabe parallel möglich bei Dateien mit Suffix |
| | *.ASF |
| | *.AVI |
| | *.MP3 |
| | *.MPEG |
| | *.WAV |
| | *.WM |
| | *.WMA |
| | *.WMV |
| currentItemChange | erzeugt, wenn ID_Player.controls.currentItem sich ändert |
| Hinweis: | aktuelles media Objekt ID_Player.currentMedia |
| | aktuelles Media Item ID_Player.controls.currentItem |
| | aktuelle Playliste ID_Player.currentPlaylist |
| | Playliste ID_Player.playlistCollection.item() |
| Syntax: | currentItemChange() |
| currentPlaylistChange | erzeugt, wenn ID_Player.currentPlaylist sich ändert |
| Hinweis: | aktuelles media Objekt ID_Player.currentMedia |
| | aktuelles Media Item ID_Player.controls.currentItem |
| | aktuelle Playliste ID_Player.currentPlaylist |
| | Playliste ID_Player.playlistCollection.item() |
| Syntax: | currentPlaylistChange(change) |
| | change Integer |
| | 0 undefiniert |
| | 1 Playliste ist leer |
| | 2 Playlisten-Info wurde geändert |
| | 3 Playlisten-Eintrag wurde verschoben |
| | 4 Playlisten-Eintrag wurde gelöscht |
| | 5 Playliste wurde erweitert durch einfügen eines Playlisten-Eintrages |
| | 6 Playliste wurde erweitert durch anhängen eines Playlisten-Eintrages |
| | 8 Playliste-Name wurde geändert |
| error | erzeugt, wenn irgendein Fehler auftritt |
| Syntax: | error() |
| markerHit | erzeugt, wenn ein Marker erreicht wurde |
| Syntax: | markerHit(MarkerNum) |
| | MarkerNum Integer |
| | Nummer des Markers |
| | ab 1 |
| mediaChange | erzeugt, wenn ein media Objekt bzw. Media Item sich ändert |
| Hinweis: | aktuelles media Objekt ID_Player.currentMedia |
| | aktuelles Media Item ID_Player.controls.currentItem |
| | aktuelle Playliste ID_Player.currentPlaylist |
| | Playliste ID_Player.playlistCollection.item() |
| Syntax: | mediaChange(Item) |



| | | |
|-----------------------|---|--|
| | Item | Zeiger auf media Objekt bzw. Media Item |
| mediaCollectionChange | erzeugt, wenn ein Collection mediaCollection sich ändert | |
| | Hinweis: | aktuelles media Objekt ID_Player.currentMedia
aktuelles Media Item ID_Player.controls.currentItem
aktuelle Playliste ID_Player.currentPlaylist
Playliste ID_Player.playlistCollection.item() |
| | Syntax: | mediaCollectionChange() |
| modeChange | erzeugt, wenn sich der Modus der Wiedergabe von Tracks ändert (Modus des Players) | |
| | aktuellen Modus ermitteln per ID_Player.settings.getMode() | |
| | Syntax: | modeChange(ModeName, NewValue) |
| | ModeName | String |
| | | "shuffle" Tracks ab nächsten Track in Zufallsreihenfolge wiedergeben |
| | | "loop" aktueller Track wird endlos wiedergegeben |
| | NewValue | true, so Modus laut ModeName ist aktiv
false, so Modus laut ModeName ist nicht aktiv |
| openStateChange | erzeugt, wenn sich ID_Player.openState ändert | |
| | Syntax: | openStateChange(NewState) |
| | NewState | Integer |
| | | 0 undefined
1 eine neue Playliste wird gerade geladen (auch bei Netzwerk)
2 Player sucht Playliste
Playliste kann sein
lokal oder im Netzwerk
Datenbank oder Textdatei
3 Player verbindet sich mit der Playliste (auch bei Netzwerk)
4 Playliste wurde gefunden und es wird nun auf sie zugegriffen (auch bei Netzwerk)
5 Playliste wurde gefunden und wird nun ausgelesen (auch bei Netzwerk)
6 Playliste ist offen (auch bei Netzwerk)
7 neue Playliste wurde zur aktuellen Playliste
8 eine neue Media-Datei wird gerade geladen (auch bei Netzwerk)
9 Player sucht Media-Datei
Media-Datei kann sein
lokal oder im Netzwerk
10 Player verbindet sich mit Server, der die Media-Datei hat (nur bei Netzwerk)
11 Media-Datei wurde gefunden und es wird nun auf sie zugegriffen (auch bei Netzwerk)
12 Media-Datei wurde gefunden und sie wird nun geöffnet (auch bei Netzwerk)
13 Media-Datei ist offen (auch bei Netzwerk)
14 Codec-Ermittlung wird gestartet
15 Codec-Ermittlung ist komplett und beendet
Codec ist gültig
16 Lizenz-Ermittlung wird gestartet
Lizenz bei DRM protected Medium
17 Lizenz-Ermittlung ist komplett und beendet
Lizenz ist gültig
18 Individualisierung wird gestartet
DRM-Individualisierung
19 Individualisierung ist komplett und beendet
20 Player wartet auf ein Medium
21 Medium-Datei hat unbekannten Typ |
| playlistChange | erzeugt, wenn sich eine Playliste ändert | |
| | Syntax: | |



playListChange(Playlist, change)

Playlist Zeiger auf Playliste (kann die aktuelle Playliste sein, muss nicht)

change Integer

- 0 undefiniert
- 1 Playliste ist leer
- 2 Playlisten-Info wurde geändert
- 3 Playlisten-Eintrag wurde verschoben
- 4 Playlisten-Eintrag wurde gelöscht
- 5 Playliste wurde erweitert durch Einfügen eines Playlisten-Eintrages
- 6 Playliste wurde erweitert durch Anhängen eines Playlisten-Eintrages
- 8 Playliste-Name wurde geändert

playStateChange
Wiedergabe)

erzeugt, wenn sich der Status der Wiedergabe der aktuellen Media-Datei (Status des Players bei

ändert

Media-Datei muss bestimmte Wiedergabe-Möglichkeiten unterstützen, damit deren Status angezeigt wird

Download und Wiedergabe parallel möglich bei Dateien mit Suffix

*.ASF

*.AVI

*.MP3

*.MPEG

*.WAV

*.WM

*.WMA

*.WMV

Syntax:

playStateChange(NewState)

NewState Integer

- 0 undefined
- 1 Wiedergabe ist gestoppt
- 2 Wiedergabe pausiert
- 3 Wiedergabe erfolgt fortlaufend
- 4 Wiedergabe mit schnellem Vorlauf
- 5 Wiedergabe mit schnellem Rücklauf
- 6 Media-Datei wird vom Server gelesen und gepuffert
Download und Wiedergabe parallel möglich bei
Dateien mit Suffix
 - *.ASF
 - *.AVI
 - *.MP3
 - *.MPEG
 - *.WAV
 - *.WM
 - *.WMA
 - *.WMV
- 7 Player wartet auf Antwort vom Server für
Beginn einer Sitzung
- 8 Wiedergabe ist komplett und beendet
- 9 es wird gerade eine neue Media-Datei vorbereitet
für Wiedergabe
- 10 Wiedergabe kann beginnen, da Player bereit dafür ist

positionChange

erzeugt, wenn die aktuelle Position im Medium eine andere ist als eine Vergleichsposition

Syntax:

positionChange(oldPosition, newPosition)

oldPosition

Integer

newPosition

Integer

scriptCommand

erzeugt, wenn ein synchronisiertes Kommando oder eine URL aus einer Media-Datei erkannt wurde
z.B. aus einer *.asf-Datei

Die Scriptkommandos der Datei sind Teil des Media-Datenstromes und steuern während der
Medium-Wiedergabe den Windows Media Player.

Desweiteren kann zusätzlich Javascript oder VBScript und HTML eingebunden werden für eine
individuelle Animation zum Player.

ASF-Datei ist Text-Datei, die je 2 Zeilen für Scriptkommandos besitzt:

z.B.




```
scType="URL"
```

```
Param=http://www.test.de/start.htm&&frame_name
```

mit frame_name als Name eines Frames in start.htm

Die erste Zeile ist der Kommandotyp also **scType**="kette_in_grossbuchstaben"

Kodierung von " " ist **zwingend**

Die zweite Zeile ist der Kommandowert also **Param** = kette_mit_freiem_inhalt

keine Kodierung von " "

Syntax:

```
scriptCommand(scType, Param)
```

scType String

Typ des Script-Kommandos

anhand Typ weiss der Player, wie der Wert laut Param zu

verarbeiten ist

z.B.

"URL"

Wert laut Param ist Url eines HTML-Dokumentes
bewirkt sofortigen Laden des Dokumentes in den Standard-Webbrowser, wenn player.settings.invokeURLs auf true steht (sonst kein laden)

ID_Player.settings.baseURL wird ignoriert

Standardname des Frame laut

ID_Player.settings.defaultFrame

"FILENAME"

Wert laut Param ist Url einer Media-Datei

"TEXT"

Wert laut Param ist Caption (Text), der angezeigt wird

Param Wert des Scriptkommandos

z.B. Wert ist eine Url, wenn scType="URL"

Beispiel:

```
<SCRIPT FOR="ID_Player"
  EVENT="ScriptCommand(scType, Param)"
  LANGUAGE="JScript"
>
  if (scType.toLowerCase() == "url")
  {document.all.zeiger_auf_objekt.innerHTML = Param; }
</SCRIPT>
```

Hinweis: zeiger_auf_objekt laut ID-Attribut des HTML-Elementes
Die Zuweisung auf .innerHTML löst sofortiges Parsen aus.
scType und Param sind Schlüsselworte zum Event ScriptCommand

statusChange erzeugt, wenn ID_Player.status sich verändert

Syntax:

```
statusChange()
```

5.2.4.5.2. ID_Player.cdromCollection Collection

Collection zur Verwaltung von CD-Laufwerken.

Ein Element der Collection ist genau ein Objekt, das **genau** 1 CD-Laufwerk verwaltet.

Syntax:

Collection:

```
ID_Player.cdromCollection.eigenschaft
ID_Player.cdromCollection.methode()
```

Element (CD-Laufwerk):

ohne CD-Laufwerksbuchstabe:

```
[ var Zeiger = ] ID_Player.cdromCollection.item(index)
```

index

Index in der Collection cdromCollection

Integer

ab 0



mit CD-Laufwerksbuchstabe:

```
[ var Zeiger = ] ID_Player.cdromCollection.getByDriveSpecifler(Kette)
```

Kette String
Laufwerksbuchstabe

Eigenschaften:

.count Anzahl der im System verfügbaren CD-Laufwerke
Anzahl der Elemente in der Collection
Achtung: **nicht** .length
.driveSpecifler liefert Laufwerksbuchstabe des CD-Laufwerkes als Element der Collection
.playlist Zeiger auf Playliste auf einer CD im CD-Laufwerk
aktuelle Playliste: siehe Objekt ID_Player.currentPlaylist

Methoden:

.eject() CD-Laufwerk als Element der Collection öffnen
ohne Laufwerksbuchstabe
Achtung: CD darf nicht gerade wiedergegeben werden, also vorher ID_Player.playState prüfen
.item() Zeiger auf ein CD-Laufwerk als Element der Collection liefern:
ohne Laufwerksbuchstabe
.getByDriveSpecifler() Zeiger auf ein CD-Laufwerk als Element der Collection liefern:
mit Laufwerksbuchstabe

5.2.4.5.3. ID_Player.closedCaption Objekt

dient zum Einschliessen von Captions (Text) in ein media Objekt
siehe auch ID_Player und dort Event scriptCommand

Syntax:

ID_Player.closedCaption.eigenschaft

Eigenschaften:

.captioningID Name des Frame oder des Control, in denen das Caption angezeigt wird
.SAMIFileName Name der Synchronized Accessible Media Interchange (SAMI)-Datei, die Informationen zum Caption enthält
Suffix der Datei *.smi oder *.sami
.SAMILang Sprache der der Synchronized Accessible Media Interchange (SAMI)-Datei, die Informationen zum Caption enthält
Suffix der Datei *.smi oder *.sami
Sprache liegt in der Datei innerhalb <STYLE> .. </STYLE>
z.B. .ENUSCC für US-Englisch
Sprache ist alphanumerisch und muss mit Punkt beginnen
mehrere Sprachen möglich
.SAMISyle Style des Caption
Sprache liegt in der Datei innerhalb <STYLE> .. </STYLE>
ist Kette mit erstem Zeichen stets das Nummernkreuz #
Bsp. #BigFont

Methoden:

keine

5.2.4.5.4. ID_Player.controls Objekt

dient zur Steuerung der Wiedergabe der aktuellen Media-Datei / aktuellen Media-Item

Playliste: mindestens 2 Einträgen aus media Objekten
Einträge werden Media Item benannt

Hinweise: ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart
wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu ID_Player.URL
ID_Player.currentMedia
setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem()
ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das nächste



Playlisten-Eement aktiviert.
 Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.
 Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger gesetzt !

Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist:
 Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

| | |
|---|---|
| Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei | ID_Player.URL |
| media Objekt erzeugen aus geladenener Media-/Meta-Datei | ID_Player.launchURL |
| Media-Datei laden und als media Objekt der Bibliothek hinzufügen | ID_Player.mediaCollection.add() |
| media Objekt aktuell setzen | ID_Player.currentMedia = zeiger_auf_media_objekt; |
| aktuelles media Objekt | ID_Player.currentMedia |
| prüfen ob aktuelles media Objekt eine Media Item ist, | |
| also einer Playliste angehört | ID_Player.currentMedia.isMemberOf() |
| Playliste | ID_Player.playlistCollection.item() |
| Playliste mit Namen erzeugen | ID_Player.playlistCollection.newPlaylist() |
| Playliste in die Media-Bibliothek importieren | ID_Player.playlistCollection.importPlaylist() |
| Playliste aus der Media-Bibliothek entfernen | ID_Player.playlistCollection.remove() |
| Playliste aus allen Elemente der Media-Bibliotheks erzeugen | ID_Player.mediaCollection.getAll() |
| Playliste auf einer CD im CD-Laufwerk | ID_Player.cdromCollection.item().playlist |
| Playliste als aktuell setzen | ID_Player.currentPlaylist = zeiger_auf_playliste; |
| aktuelle Playliste | ID_Player.currentPlaylist |
| media Objekt an das Ende der aktuellen Playliste anhängen und | |
| damit dort einen Media-Item erzeugen | ID_Player.currentPlaylist.appendItem() |
| Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe: | ID_Player.cdromCollection.item() |
| Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe: | ID_Player.cdromCollection.getByDriveSpecifier() |
| aktuell per Player-Control manipulierbares media Objekt bzw. Media Item | ID_Player.controls.currentItem |
| Verfügbarkeit von Controls zu einer Media-Datei / Media-Item | ID_Player.controls.isAvailable() |
| nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste anwählen | ID_Player.controls.next() |
| Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe | ID_Player.controls.playItem() |
| vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste | |
| anwählen | ID_Player.controls.previous() |
| Anzahl der Wiedergaben (nicht der Wiederholungen) | ID_Player.settings.playCount |
| Wiedergabe-Modus (zufall, endlos) | ID_Player.settings.setMode() |
| Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung | |
| zum Mediumtyp) | ID_Player.settings.volume |

Syntax:

ID_Player.controls.eigenschaft
 ID_Player.controls.methode()

Eigenschaften:

| | |
|------------------------|--|
| .currentItem | Zeiger auf aktuelles media Objekt in einer Playliste (Media Item) |
| | media Objekt als aktuelles Element der Playliste setzen |
| .currentMarker | Nummer des aktuellen Marker in der Media-Datei |
| | Marker als aktuellen Marker setzen |
| | Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount |
| | Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei |
| .currentPosition | aktuelle Wiedergabe-Position als Wert in Sekunden in der Media-Datei ab Beginn der Datei |
| | aktuelle Wiedergabe-Position setzen |
| .currentPositionString | aktuelle Wiedergabe-Position als String in der Media-Datei ab Beginn der Datei |

Methoden:

| | |
|----------------|---|
| .fastForward() | schnelles Vorspulen während der Wiedergabe einer Media-Datei |
| | Standardgeschwindigkeit des schnellen Vorspulens während der Wiedergabe ist |
| | 5 faches der normalen Wiedergabegeschwindigkeit |
| | ansonsten per ID_Player.setting.rate einstellbar |
| | ignoriert ID_Player.settings.rate |
| | wird aufgehoben durch ID_Player.controls.play |
| | ID_Player.controls.stop |
| | hebt auf ID_Player.controls.fastReverse() |
| | nicht möglich z.B. für Audio |
| | Ermittlung, ob Medium schnelles Vorspulen bei Wiedergabe unterstützt, per |
| | ID_Player.controls.isAvailable() |
| .fastReverse() | schnelles Rückspulen während der Wiedergabe einer Media-Datei |



| | | |
|----------------|--|--|
| | Standardgeschwindigkeit des schnellen Rückspulens während der Wiedergabe ist
5 faches der normalen Wiedergabegeschwindigkeit
ansonsten per ID_Player.setting.rate einstellbar
nur innerhalb eines Track
ignoriert | ID_Player.settings.playCount
ID_Player.settings.rate |
| | wird aufgehoben durch | ID_Player.controls.fastForward()
ID_Player.controls.play
ID_Player.controls.stop |
| | nicht möglich z.B. für Audio
Ermittlung, ob Medium schnelles Rückspulen bei Wiedergabe unterstützt, per | ID_Player.controls.isAvailable() |
| .isAvailable() | Verfügbarkeit von Controls zu einer Media-Datei | |
| .next() | nächsten Eintrag (Media Item) vorwärts in der Playliste anwählen:
Wenn bereits LETZTER Eintrag der Playliste erreicht wurde und dann .next() aktiviert wird,
dann wird der ERSTE Eintrag in der Playliste eingestellt. | |
| | Medium muss Playliste unterstützen | |
| .pause() | pausieren der Wiedergabe des aktuellen media Objektes bzw. Media Items
siehe auch | ID_Player.controls.play()
ID_Player.controls.playItem()
ID_Player.controls.stop() |
| .play() | Wiedergabe des aktuellen media Objektes bzw. Media Item starten
oder wenn aktuelles Media bzw. Media Item bereits pausiert, so wird Pause aufgehoben
siehe auch | ID_Player.controls.playItem()
ID_Player.controls.pause()
ID_Player.controls.stop() |
| .playItem() | Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe
ID_Player.setting.autoStart wird ignoriert
Hinweis: beliebige Media Datei laden und wiedergeben per
per ID_Player.url
und danach ID_Player.controls.play()
siehe auch | ID_Player.controls.play()
ID_Player.controls.pause()
ID_Player.controls.stop() |
| .previous() | vorhergehenden Eintrag (Media Item) rückwärts in der Playliste anwählen:
Wenn bereits ERSTER Eintrag der Playliste erreicht wurde und dann .previous() aktiviert wird,
dann wird der LETZTE Eintrag in der Playliste eingestellt. | |
| | Medium muss Playliste unterstützen | |
| .stop() | Wiedergabe des aktuellen media Objektes bzw. Media Item (aktuelles media Objekt der Playliste) beenden
oder wenn aktuelles Media bzw. Media Item bereits pausiert, so wird Pause aufgehoben
Achtung: Die Systemressourcen zur Media-Datei werden freigegeben !
bei Track: automatisch immer auf Trackanfang gesetzt
siehe auch | ID_Player.controls.play()
ID_Player.controls.playItem()
ID_Player.controls.pause() |

5.2.4.5.5. ID_Player.currentMedia Objekt

ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu

ID_Player.URL
ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem()
ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das
nächste Playlisten-Eement aktiviert.

Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger
gesetzt !

Hinweise: Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert



wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist: Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei
media Objekt erzeugen aus geladenener Media-/Meta-Datei
Media-Datei laden und als media Objekt der Bibliothek hinzufügen
media Objekt aktuell setzen
aktuelles media Objekt
prüfen ob aktuelles media Objekt eine Media Item ist,
also einer Playliste angehört

Playliste
Playliste mit Namen erzeugen
Playliste in die Media-Bibliothek importieren
Playliste aus der Media-Bibliothek entfernen
Playliste aus allen Elemente der Media-Bibliotheks erzeugen
Playliste auf einer CD im CD-Laufwerk
Playliste als aktuell setzen
aktuelle Playliste
media Objekt an das Ende der aktuellen Playliste anhängen und
damit dort einen Media-Item erzeugen

Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe:
Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe:

aktuell per Player-Control manipulierbares media Objekt bzw. Media Item
Verfügbarkeit von Controls zu einer Media-Datei / Media-Item
nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste anwählen
Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe
vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste
anwählen
Anzahl der Wiedergaben (nicht der Wiederholungen)
Wiedergabe-Modus (zufall, endlos)
Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung
zum Mediumtyp)

ID_Player.URL
ID_Player.launchURL
ID_Player.mediaCollection.add()
ID_Player.currentMedia = zeiger_auf_media_objekt;
ID_Player.currentMedia

ID_Player.currentMedia.isMemberOf()
ID_Player.playlistCollection.item()
ID_Player.playlistCollection.newPlaylist()
ID_Player.playlistCollection.importPlaylist()
ID_Player.playlistCollection.remove()
ID_Player.mediaCollection.getAll()
ID_Player.cdromCollection.item().playlist
ID_Player.currentPlaylist = zeiger_auf_playlist;
ID_Player.currentPlaylist

ID_Player.currentPlaylist.appendItem()
ID_Player.cdromCollection.item()
ID_Player.cdromCollection.getByDriveSpecifier()

ID_Player.controls.currentItem
ID_Player.controls.isAvailable()
ID_Player.controls.next()
ID_Player.controls.playItem()

ID_Player.controls.previous()
ID_Player.settings.playCount
ID_Player.settings.setMode()

ID_Player.settings.volume

Syntax:

ID_Player.currentMedia.eigenschaft
ID_Player.currentMedia.methode()

Beispiel:

```
function Ermitteln()
{
    var Kette;
    var AktuellesAttribut_Name;
    var AktuellesAttribut_Wert;

    // Media-Bibliothek
    var MediaBibliothek = ID_Player.mediaCollection;

    // Medium hinzufügen
    var Medium = MediaBibliothek.add("test.mid");

    // Medium als aktuelles Medium setzen durch Zuweisung
    ID_Player.currentMedia = Medium;

    // Zeiger auf aktuelles Medium
    var AktuellesMedium = Medium;

    // Attribute erzeugen
    AktuellesMedium.setItemInfo("Attribut1", "wert1");
    AktuellesMedium.setItemInfo("Attribut2", "wert2");

    var AktuellesMediumAnzahlAttribute = AktuellesMedium.attributeCount;

    Kette = "Anzahl der Attribute akt. Medium = " + AktuellesMediumAnzahlAttribute + "\r\r"
    + " Attribute sind :\r\r";
```



```

// alle Attribute des aktuellen Medium abklappern
for (var AttributeZahler = 0; AttributeZahler < AktuellesMediumAnzahlAttribute; ++ AttributeZahler)
{
    // Attributname
    AktuellesAttribut_Name = AktuellesMedium.getAttributeName(AttributeZahler);

    // Attributwert
    AktuellesAttribut_Wert = AktuellesMedium.getItemInfo(AktuellesAttribut_Name);

    Kette += AktuellesAttribut_Name + ": " + AktuellesAttribut_Wert + "\r";
}

ID_Div.innerHTML= Kette;
}

<DIV ID="ID_Div">
</DIV>

```

"ID_Player" ist der Wert des ID-Attributes im OBJECT-Tag

Eigenschaften:

| | |
|--------------------|--|
| .attributeCount | Anzahl der Attribute des aktuellen media Objektes |
| .duration | zeitliche Dauer in Sekunden (Integer) des aktuellen media Objektes |
| .durationString | zeitliche Dauer in Sekunden (String) des aktuellen media Objektes |
| .imageSourceHeight | Höhe in Pixel des aktuellen media Objektes |
| .imageSourceWidth | Breite in Pixel des aktuellen media Objektes |
| .markerCount | Anzahl der Marker in der Media-Datei des aktuellen media Objektes |
| | Media-Datei muss Marker unterstützen |
| | Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei |
| | Nummer des aktuellen Marker laut ID_Player.controls.currentMarker |
| .name | Name des aktuellen media Objektes |
| .sourceURL | Url des aktuellen media Objektes |

Methoden:

| | |
|----------------------|--|
| .getAttributeName() | Name eines Attributes des aktuellen media Objektes liefern |
| .getItemInfo() | Wert eines Attributes des aktuellen media Objektes liefern |
| .getItemInfoByAtom() | Wert eines Attributes des aktuellen media Objektes liefern |
| .getMarkerName() | Name eines Markers in der Media-Datei des aktuellen media Objektes liefern |
| | Media-Datei muss Marker unterstützen |
| | Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei |
| | Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount |
| | Nummer des aktuellen Marker laut ID_Player.controls.currentMarker |
| .getMarkerTime() | Zeitpunkt eines Markers in der Media-Datei des aktuellen media Objektes liefern |
| | Media-Datei muss Marker unterstützen |
| | Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei |
| | Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount |
| | Nummer des aktuellen Marker laut ID_Player.controls.currentMarker |
| .isIdentical() | auf Identität des aktuellen media Objektes mit einem anderen (nicht aktuellen) media Objektes prüfen |
| .isMemberOf() | prüfen ob aktuelles media Objekt eine Media Item ist, also einer Playliste angehört |
| .isReadOnlyItem() | Veränderbarkeit eines Attributes des aktuellen media Objektes |
| .setItemInfo() | Wert eines Attributes des aktuellen media Objektes setzen |
| | wenn Attribut nicht vorhanden, so erzeugt |
| | Veränderbarkeit eines Attributes laut ID_Player.currentMedia.isReadOnlyItem() |

5.2.4.5.6. ID_Player.currentPlaylist Objekt

referenziert die aktuelle Playliste der Media-Bibliothek

Hinweise: ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu

ID_Player.URL
ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht



ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem() ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das nächste Playlisten-Element aktiviert.

Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger gesetzt !

Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist: Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei
media Objekt erzeugen aus geladenener Media-/Meta-Datei
Media-Datei laden und als media Objekt der Bibliothek hinzufügen
media Objekt aktuell setzen
aktuelles media Objekt
prüfen ob aktuelles media Objekt eine Media Item ist,
also einer Playliste angehört

Playliste
Playliste mit Namen erzeugen
Playliste in die Media-Bibliothek importieren
Playliste aus der Media-Bibliothek entfernen
Playliste aus allen Elemente der Media-Bibliotheks erzeugen
Playliste auf einer CD im CD-Laufwerk
Playliste als aktuell setzen
aktuelle Playliste
media Objekt an das Ende der aktuellen Playliste anhängen und
damit dort einen Media-Item erzeugen

Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe:
Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe:

aktuell per Player-Control manipulierbares media Objekt bzw. Media Item
Verfügbarkeit von Controls zu einer Media-Datei / Media-Item
nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste anwählen
Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe
vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste
anwählen
Anzahl der Wiedergaben (nicht der Wiederholungen)
Wiedergabe-Modus (zufall, endlos)
Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung
zum Mediumtyp)

ID_Player.URL
ID_Player.launchURL
ID_Player.mediaCollection.add()
ID_Player.currentMedia = zeiger_auf_media_objekt;
ID_Player.currentMedia

ID_Player.currentMedia.isMemberOf()

ID_Player.playlistCollection.item()
ID_Player.playlistCollection.newPlaylist()
ID_Player.playlistCollection.importPlaylist()
ID_Player.playlistCollection.remove()
ID_Player.mediaCollection.getAll()
ID_Player.cdromCollection.item().playlist
ID_Player.currentPlaylist = zeiger_auf_playlist;
ID_Player.currentPlaylist

ID_Player.currentPlaylist.appendItem()

ID_Player.cdromCollection.item()
ID_Player.cdromCollection.getByDriveSpecifier()

ID_Player.controls.currentItem
ID_Player.controls.isAvailable()
ID_Player.controls.next()
ID_Player.controls.playItem()

ID_Player.controls.previous()
ID_Player.settings.playCount
ID_Player.settings.setMode()

ID_Player.settings.volume

Syntax:

ID_Player.currentPlaylist.eigenschaft
ID_Player.currentPlaylist.methode()

Beispiel:

```
function Ermitteln()
{
    var Kette;
    var AktuellesAttribut_Name;
    var AktuellesAttribut_Wert;

    // aktuelle Playliste
    var AktuellePlayListe = ID_Player.currentPlaylist;

    // Playlist-Attribut erzeugen
    AktuellePlayListe.setItemInfo("PlaylistAttribut1", "changed");

    // Playlisteintrag mit Index 0 erweitern mit neuem Attribut
    AktuellePlayListe.item(0).setItemInfo("PlaylistAttribut2", "5");

    var AnzahlPlayListAttribute = AktuellePlayListe.attributeCount;
```




```

Kette = "Anzahl der Attribute der Playliste = " + AnzahlPlayListAttribute + "\r\r"
      + " Playlist-Attributes sind :\r\r";

// alle Attribute der Playliste abklappen
for (var AttributeZahler = 0; AttributeZahler < AnzahlPlayListAttribute; ++ AttributeZahler)
{
    // Attributname
    AktuellesAttribut_Name = AktuellePlayListe.attributeName(AttributeZahler);

    // Attributwert
    AktuellesAttribut_Wert = AktuellePlayListe.getItemInfo(AktuellesAttribut_Name);

    Kette += AktuellesAttribut_Name + ": " + AktuellesAttribut_Wert + "\r\r";
}

// alle Playlisten-Einträge abklappen
var AnzahlPlayListEintraege = AktuellePlayListe.count;

for ( var PlayListEintragZahler = 0;
      PlayListEintragZahler < AnzahlPlayListEintraege;
      ++ PlayListEintragZahler
    )
{
    var PlayListEintrag = AktuellePlayListe.item(PlayListEintragZahler);
    var PlayListEintrag_AnzahlAttribute = PlayListEintrag.attributeCount;

    Kette += "\rPlaylisteneintrag " + PlayListEintragZahler + "\r"
          + " mit " + PlayListEintrag_AnzahlAttribute + " Attributen:\r\r";

    // alle Attribute zum Playlisteintrag abklappen
    for ( var AttributeZahler = 0;
          AttributeZahler < PlayListEintrag_AnzahlAttribute;
          ++ AttributeZahler
        )
    {
        AktuellesAttribut_Name = PlayListEintrag.getAttributeName(AttributeZahler);
        AktuellesAttribut_Wert = PlayListEintrag.getItemInfo(AktuellesAttribut_Name);

        Kette += "Attribut " + AktuellesAttribut_Name
              + " mit Wert = " + AktuellesAttribut_Wert + "\r";
    }
}

ID_Div.innerHTML= Kette;
}

<DIV ID="ID_Div">
</DIV>

```

"ID_Player" ist der Wert des ID-Attributes im OBJECT-Tag

Eigenschaften:

| | |
|-----------------|---|
| .attributeCount | Anzahl der Attribute der aktuellen Playliste |
| .count | Anzahl der Media Item's in der aktuellen Playliste |
| .name | Name der aktuellen Playliste |
| | Jede Playliste hat in der Media-Bibliothek einen eindeutigen Namen. |

Methoden:

| | |
|------------------|--|
| .appendItem() | media Objekt an das Ende der aktuellen Playliste anhängen, also dort Media-Item erzeugen
aktuelle Playliste erweitern |
| .attributeName() | Name eines Attributes der aktuellen Playliste liefern |
| .getItemInfo() | Wert eines Attributes eines Media Item's in der aktuellen Playliste liefern |
| .insertItem() | media Objekt in die aktuelle Playliste einfügen, also dort Media Item erzeugen
aktuelle Playliste erweitern
vor dem Einfügen: Media Item am Einfügeplatz und seine Nachfolger rutschen automatisch 1 Platz
weiter runter in der Playliste |
| .isIdentical() | es wird ID_Player.currentPlaylist.count erhöht um Wert 1
auf Identität der aktuellen Playliste mit einer anderen (nicht aktuellen) Playliste prüfen |
| .item() | Zeiger auf ein Media Item in der aktuellen Playliste liefern
per Zeiger kann Media Item wie ein media Objekt behandelt werden |
| .moveItem() | Media Item in der aktuellen Playliste verschieben durch Indextausch
aktuelle Playliste verändern |
| .removeItem() | Media Item aus der aktuellen Playliste entfernen
aktuelle Playliste bereinigen
Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das
nächste Playlisten-Eement aktiviert. |



Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird

ID_Player.currentMedia auf null-Zeiger gesetzt !

.setItemInfo()

Wert eines Attributes eines Media-Item's in der aktuellen Playliste setzen

wenn Attribute nicht vorhanden, so erzeugt

es besteht kein Schreibschutz für Attribute

5.2.4.5.7. ID_Player.error Objekt

Fehlerinformationen zu media Objekten bzw. Media Item's

Fehlerinformationen werden in ihrer Reihenfolge des Auftretens **per Zeiger** in einer Queue als Zeigerfeld gesammelt.

Syntax:

ID_Player.error.eigenschaft

ID_Player.error.methode()

ID_Player.error.item().eigenschaft

ID_Player.error ist ein error Objekt und enthält genau 1 Fehlerinformation

ID_Player.error.item() ist ein Eintrag aus dem Feld (Queue) der Zeiger aller error Objekte

ist ein Zeiger auf ein error Objekt

Fehlerinformationen werden in ihrer Reihenfolge des Auftretens in einer Queue als Zeigerfeld gesammelt.

Eigenschaften:

.errorCount Anzahl der Fehlerinformationen in der Fehler-Queue

.errorDescription Fehlertext (String)

Methoden:

.clearErrorQueue() Fehler-Queue löschen, also alle Fehlerinformationen zu allen media Objekten / Media Item's löschen

.item() Zeiger auf Fehler (errorItem Objekt) in der Fehler-Queue liefern

.webHelp() Hilfe-Seite von Microsoft im Internet zum Windows Media Player aktivieren

Auf der Hilfe-Seite stehen Informationen zu Fehlern.

5.2.4.5.8. ID_Player.mediaCollection Collection

eingeschränkte Verwaltung der Media-Bibliothek bezüglich ihrer media Objekte (nicht Media Item's) und vorallem Playlisten

Media-Bibliothek wird ansonsten vom Player verwaltet und gespeichert

für Feld der Zeiger aller Playlisten der Media-Bibliothek: siehe playlistCollection Collection

Hinweise: ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu

ID_Player.URL

ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu

ID_Player.URL

ID_Player.currentMedia

durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu ID_Player.URL

ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem() ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das nächste Playlisten-Element aktiviert.

Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger gesetzt !

Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist: Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei

ID_Player.URL



media Objekt erzeugen aus geladenener Media-/Meta-Datei
 Media-Datei laden und als media Objekt der Bibliothek hinzufügen
 media Objekt aktuell setzen
 aktuelles media Objekt
 prüfen ob aktuelles media Objekt eine Media Item ist,
 also einer Playliste angehört

Playliste
 Playliste mit Namen erzeugen
 Playliste in die Media-Bibliothek importieren
 Playliste aus der Media-Bibliothek entfernen
 Playliste aus allen Elemente der Media-Bibliotheks erzeugen
 Playliste auf einer CD im CD-Laufwerk
 Playliste als aktuell setzen
 aktuelle Playliste
 media Objekt an das Ende der aktuellen Playliste anhängen und
 damit dort einen Media-Item erzeugen

Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe:
 Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe:

aktuell per Player-Control manipulierbares media Objekt bzw. Media Item
 Verfügbarkeit von Controls zu einer Media-Datei / Media-Item
 nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste anwählen
 Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe
 vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste
 anwählen
 Anzahl der Wiedergaben (nicht der Wiederholungen)
 Wiedergabe-Modus (zufall, endlos)
 Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung
 zum Mediumtyp)

ID_Player.launchURL
 ID_Player.mediaCollection.add()
 ID_Player.currentMedia = zeiger_auf_media_objekt;
 ID_Player.currentMedia
 ID_Player.currentMedia.isMemberOf()
 ID_Player.playlistCollection.item()
 ID_Player.playlistCollection.newPlaylist()
 ID_Player.playlistCollection.importPlaylist()
 ID_Player.playlistCollection.remove()
 ID_Player.mediaCollection.getAll()
 ID_Player.cdromCollection.item().playlist
 ID_Player.currentPlaylist = zeiger_auf_playlist;
 ID_Player.currentPlaylist
 ID_Player.currentPlaylist.appendItem()
 ID_Player.cdromCollection.item()
 ID_Player.cdromCollection.getByDriveSpecifier()
 ID_Player.controls.currentItem
 ID_Player.controls.isAvailable()
 ID_Player.controls.next()
 ID_Player.controls.playItem()
 ID_Player.controls.previous()
 ID_Player.settings.playCount
 ID_Player.settings.setMode()
 ID_Player.settings.volume

Syntax:

ID_Player.mediaCollection.methode()
 ID_Player.mediaCollection.getAttributeStringCollection().eigenschaft
 ID_Player.mediaCollection.getAttributeStringCollection().methode()

Beispiel:

```
function Ermitteln()
{
    var Kette;
    var AktuellesAttribut_Name;
    var AktuellesAttribut_Wert;

    // Media-Bibliothek
    var MediaBibliothek = ID_Player.mediaCollection;

    // Medium hinzufügen
    var Medium = MediaBibliothek.add("test.mid");

    // Medium als aktuelles Medium setzen durch Zuweisung
    ID_Player.currentMedia = Medium;

    // Zeiger auf aktuelles Medium
    var AktuellesMedium = Medium;

    // Attribute erzeugen
    AktuellesMedium.setItemInfo("Attribut1", "wert1");
    AktuellesMedium.setItemInfo("Attribut2", "wert2");

    var AktuellesMediumAnzahlAttribute = AktuellesMedium.attributeCount;

    Kette = "Anzahl der Attribute akt. Medium = " + AktuellesMediumAnzahlAttribute + "\r\r"
    + " Attribute sind :\r\r";

    // alle Attribute des aktuellen Medium abklappern
    for (var AttributeZahler = 0; AttributeZahler < AktuellesMediumAnzahlAttribute; ++ AttributeZahler)
    {
        // Attributname
        AktuellesAttribut_Name = AktuellesMedium.getAttributeName(AttributeZahler);

        // Attributwert
        AktuellesAttribut_Wert = AktuellesMedium.getItemInfo(AktuellesAttribut_Name);

        Kette += AktuellesAttribut_Name + ": " + AktuellesAttribut_Wert + "\r";
    }
}
```



```

    }

    ID_Div.innerHTML= Kette;
}

<DIV ID="ID_Div">
</DIV>

```

"ID_Player" ist der Wert des ID-Attributes im OBJECT-Tag

Eigenschaften:

.count Anzahl der Elemente im Zeichenkettenfeld laut ID_Player.mediaCollection.getAttributeStringCollection()

Methoden:

.add() Media-Datei laden und als media Objekt der Bibliothek hinzufügen und Zeiger liefern

.getAll() Playliste aus allen Elemente der Media-Bibliothek erzeugen und Zeiger liefern

.getAttributeStringCollection() je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

Zeichenkettenfeld aus allen Werte **eines** Attributes zu einem Media-Typ erzeugen

pro media Objekt vom Media-Typ wird ein Element in der Collection erzeugt

media Objekt nur dann verwendet, wenn auch Attribut mit ihm verbunden ist

Bsp.: für Attribut:

"Album"

"Author"

Bsp.: für Media-Typ

"Audio"

"Video"

.getByAlbum() Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Album"

und darin einen **bestimmten** Wert besitzen, und Zeiger liefern

je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

.getByAttribute() Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die **ein** Attribut mit **bestimmten**

Wert besitzen, und Zeiger liefern

je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

.getByAuthor() Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Author" und darin

einen **bestimmten** Wert besitzen, und Zeiger liefern

je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

.getByGenre() Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Genre" und darin

einen **bestimmten** Wert besitzen, und Zeiger liefern

je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

.getByName() Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Name" und darin

einen **bestimmten** Wert besitzen, und Zeiger liefern

je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode

.getMediaAtom() Index eines Attributes liefern, also Art des Attributes

.isDeleted() prüfen ob Media Datei aus dem Datei-Pool der Bibliothek entfernt wurde

bei lokaler Media Datei: den Papierkorb des Systems prüfen

bei nicht-lokaler Datei: auf logische Verknüpfung prüfen

Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen:

ID_Player.mediaCollection.setDeleted()

Media-Datei aus ID_Player.mediaCollection. entfernen und optional aus der Datenbank der Namen:

ID_Player.mediaCollection.remove()

.item() Wert eines Elementes im Zeichenkettenfeld laut ID_Player.mediaCollection.getAttributeStringCollection()

ermitteln

.remove() media Objekt aus Media-Bibliothek, also aus ID_Player.mediaCollection, entfernen:

aus Datenbank der Namen der Objekte und **optional** aus dem Datei-Pool

Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen: ID_Player.mediaCollection.setDeleted()

.setDeleted() Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen:

bei lokaler Media Datei: **immer** in den Papierkorb des Systems verschieben

bei nicht-lokaler Datei: logische Verknüpfung entfernen

5.2.4.5.9. ID_Player.network Objekt

Objekt für Netzwerkverbindung/Internetverbindung des Windows Media Player z.B. per Proxy

Syntax:

ID_Player.network.eigenschaft

ID_Player.network.methode()

Eigenschaften:

.bandWidth aktuelle Bandbreite (Bits pro Sekunde) der Media-Datei mit Datenstrom also Video

.bitRate aktuelle Bitrate der Media-Datei

.bufferingCount Anzahl der Zeitpuffer während der Wiedergabe des aktuellen media Objektes mit Datenstrom,

also Video

.bufferingProgress aktueller Prozentanteil der Pufferung während der Wiedergabe des aktuellen media Objektes

mit Datenstrom, also Video

Fortgeschrittenheit der Pufferung

100% Pufferung entspricht komplett gepuffert

.bufferingTime Anzahl der Millisekunden als Zeit für Pufferung vor der Wiedergabe der Media-Datei

.downloadProgress aktueller Prozentanteil des Downloads des aktuellen media Objektes von einem Webserver

Fortgeschrittenheit des Download

100% Download entspricht komplett geladen



Download und Wiedergabe parallel möglich bei Dateien mit Suffix

*.ASF
*.AVI
*.MP3
*.MPEG
*.WAV
*.WM
*.WMA
*.WMV

| | |
|----------------------------|---|
| .encodedFrameRate | Framerate (Frames pro Sekunden) der Media-Datei als Video
Framerate laut Hersteller des Video
nicht aktuelle Framerate des Video (siehe ID_Player.network.FrameRate) |
| .FrameRate | aktuelle Framerate (Frames pro Sekunden) der Media-Datei als Video
nicht Framerate laut Hersteller des Video (siehe ID_Player.network.encodedFrameRate) |
| .framesSkipped
Objektes | aktuelle Anzahl der bisher wiedergegebenen Frames ab Beginn der Wiedergabe des aktuellen media
als Video |
| .lostPackets | aktuelle Anzahl der bisher verlorengegangenen Datenpakete ab Beginn der Wiedergabe des aktuellen media
Objektes mit Datenstrom, also Video
Verlust an Datenpaketen nicht bei HTTP möglich
hängt von der Netzwerkverbindung ab |
| .maxBandwidth | maximale Bandbreite (Bits pro Sekunde) für die Wiedergaben der Media-Datei mit Datenstrom also Video
besonders bei einem Datenfluss verwenden, der mit verschiedenen Bitraten ausgestattet ist
(MBS Multiple streams width different bit rates)
z.B. Herabsetzung der maximalen Bitrate wegen der schlechteren Netzwerkqualität, weil
Media-Datei mindestens eine höhere Bandbreite, als die Verbindung Server zum
Client (Player) physisch leisten kann |
| .receivedPackets | aktuelle Anzahl der bisher nicht verlorengegangenen Datenpakete ab Beginn der Wiedergabe des aktuellen
media Objektes mit Datenstrom, also Video
Verlust an Datenpaketen nicht bei HTTP möglich
hängt von der Netzwerkverbindung ab |
| .receptionQuality | aktuelle Prozentzahl der in den letzten 30 Sekunden nicht verlorengegangenen Datenpakete während der
Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video,
gegenüber der Gesamtanzahl aller in den letzten 30 Sekunden von der Media-Datei abgesendeten
Datenpakete, also inklusive der verlorenen Datenpakete.
Verlust an Datenpaketen nicht bei HTTP möglich
hängt von der Netzwerkverbindung ab |
| .recoveredPackets | aktuelle Anzahl der bisher verlorengegangenen aber wiederhergestellten Datenpakete ab Beginn der
Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video
Verlust an Datenpaketen nicht bei HTTP möglich
hängt von der Netzwerkverbindung ab |
| .sourceProtocol | Datentransport-Protokoll während der Wiedergabe des media Objektes |
| Methoden: | |
| .getProxyBypassForLocal() | prüfen, ob Proxyserver übergangen wird, wenn der Server als Media-Datei-Quelle im lokalen Netzwerk
liegt |
| .getProxyExceptionList() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Ausnahmeliste des Proxyservers liefern bezüglich
Computer
Domains
IP's
die den Proxyserver übergehen |
| .getProxyName() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Name des benutzten Proxyservers liefern |
| .getProxyPort() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Nummer des benutzten Ports des Proxyservers liefern |
| .getProxySettings() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Einstellungen des Proxyservers liefern |
| .setProxyBypassForLocal() | einstellen, ob Proxyserver übergangen wird, wenn der Server als Media-Datei-Quelle im lokalen Netzwerk
liegt |
| .setProxyExceptionList() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Ausnahmeliste des Proxyservers setzen bezüglich
Computer
Domains
IP's
die den Proxyserver übergehen |
| .setProxyName() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Name des benutzten Proxyservers setzen |
| .setProxyPort() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Nummer des benutzten Ports des Proxyservers setzen |
| .setProxySettings() | nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert
Einstellungen des Proxyservers setzen |

5.2.4.5.10. ID_Player.playlistCollection Collection

Feld der Zeiger aller Playlisten der Media-Bibliothek

für Verwaltung der Media-Bibliothek bezüglich ihrer media Objekte (nicht Media Item's) und Playlisten: siehe mediaCollection Collection



Hinweise: ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu

ID_Player.URL
ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem() ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das nächste Playlisten-Element aktiviert.

Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger gesetzt !

Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist: Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei
media Objekt erzeugen aus geladenener Media-/Meta-Datei
Media-Datei laden und als media Objekt der Bibliothek hinzufügen
media Objekt aktuell setzen
aktuelles media Objekt
prüfen ob aktuelles media Objekt eine Media Item ist,
also einer Playliste angehört

Playliste

Playliste mit Namen erzeugen

Playliste in die Media-Bibliothek importieren

Playliste aus der Media-Bibliothek entfernen

Playliste aus allen Elemente der Media-Bibliotheks erzeugen

Playliste auf einer CD im CD-Laufwerk

Playliste als aktuell setzen

aktuelle Playliste

media Objekt an das Ende der aktuellen Playliste anhängen und
damit dort einen Media-Item erzeugen

Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe:

Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe:

aktuell per Player-Control manipulierbares media Objekt bzw. Media Item
Verfügbarkeit von Controls zu einer Media-Datei / Media-Item
nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste auswählen
Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe
vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste
auswählen

Anzahl der Wiedergaben (nicht der Wiederholungen)

Wiedergabe-Modus (zufall, endlos)

Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung
zum Mediumtyp)

ID_Player.URL

ID_Player.launchURL

ID_Player.mediaCollection.add()

ID_Player.currentMedia = zeiger_auf_media_objekt;

ID_Player.currentMedia

ID_Player.currentMedia.isMemberOf()

ID_Player.playlistCollection.item()

ID_Player.playlistCollection.newPlaylist()

ID_Player.playlistCollection.importPlaylist()

ID_Player.playlistCollection.remove()

ID_Player.mediaCollection.getAll()

ID_Player.cdromCollection.item().playlist

ID_Player.currentPlaylist = zeiger_auf_playlist;

ID_Player.currentPlaylist

ID_Player.currentPlaylist.appendItem()

ID_Player.cdromCollection.item()

ID_Player.cdromCollection.getByDriveSpecifier()

ID_Player.controls.currentItem

ID_Player.controls.isAvailable()

ID_Player.controls.next()

ID_Player.controls.playItem()

ID_Player.controls.previous()

ID_Player.settings.playCount

ID_Player.settings.setMode()

ID_Player.settings.volume

Syntax:

ID_Player.playlistCollection.eigenschaft

ID_Player.playlistCollection.methode()



ID_Player.playlistCollection.getAll().eigenschaft
ID_Player.playlistCollection.getAll().methode()

Hinweis: ID_Player.playlistCollection.getAll() liefert Zeiger auf das Feld der Zeiger **aller** Playlisten in der Media-Bibliothek

Eigenschaften:

.count Anzahl der Playlisten in der Media-Bibliothek

Methoden:

.getAll() Zeiger auf Feld **aller** Playlisten in der Media-Bibliothek liefern
.getByName() Zeiger auf Feld aus maximal 1 Playliste aus der Media-Bibliothek liefern
jede Playliste hat einen eindeutigen Namen: Feld kann also maximal 1 Element besitzen
siehe auch .getAll()
.importPlaylist() Playliste in die Media-Bibliothek importieren
.isDeleted() prüfen ob Playliste im Papierkorb des Systems liegt
.item() Zeiger auf eine Playliste in der Media-Bibliothek
.newPlaylist() Playliste mit Name als Objekt instanzieren und Zeiger liefern, aber nicht in die Media-Bibliothek importieren
Jede Playliste muss einen eindeutigen Namen haben:
Es wird Fehler erzeugt, wenn Name bereits in der Media-Bibliothek vorhanden ist.
.remove() Playliste aus der Media-Bibliothek entfernen
.setDeleted() Playliste aus der Media-Bibliothek in den Papierkorb des System verschieben

5.2.4.5.11. ID_Player.settings Objekt

Einstellungen des Players für die Wiedergabe des jeweiligen aktuellen media Objektes bzw. Media Item's

Einstellungen gelten für alle Wiedergaben

Änderung der Einstellungen der Wiedergabe **während aktiver** Wiedergabe theoretisch möglich

Viele Eigenschaften vom ID_Player.settings Objekt entsprechen den PARAM's im OBJECT-Tag.

Hinweise: ID_Player.currentMedia referenziert das aktuell **wiedergegebene** media Objekt, das aus eine Playliste stammen kann (Media Item).

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu

ID_Player.URL
ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

ID_Player.currentMedia kann automatisch auf null-Zeiger gesetzt werden, wenn per ID_Player.currentPlaylist.removeItem()
ein Playlist-Element aus der aktuellen Playliste entfernt wird:

Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das
nächste Playlisten-Element aktiviert.

Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert.

Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger
gesetzt !

Media-Item ist ein spezielles media Objekt, das aus einer Playliste stammt und per ID_Player.currentPlaylist Objekt referenziert wird.

In der Playliste wird anstelle media Objekt der Begriff Media Item verwendet.

ID_Player.currentMedia.media Objekt umfasst also nicht die Eigenschaften und Methoden zu Media Item als aktuellem
Playlisten-Eintrag.

Dafür ähneln sich die Objekte ID_Player.currentMedia und ID_Player.currentPlaylist: Sie haben z.B. gemeinsame Methoden, die sich jedoch auf verschiedene Objekte beziehen.

Hinweise: media Objekt erzeugen aus geladenener Media-/Meta-Datei
media Objekt erzeugen aus geladenener Media-/Meta-Datei
Media-Datei laden und als media Objekt der Bibliothek hinzufügen
media Objekt aktuell setzen
aktuelles media Objekt
prüfen ob aktuelles media Objekt eine Media Item ist,
also einer Playliste angehört

ID_Player.URL
ID_Player.launchURL
ID_Player.mediaCollection.add()
ID_Player.currentMedia = zeiger_auf_media_objekt;
ID_Player.currentMedia
ID_Player.currentMedia.isMemberOf()



| | |
|--|---|
| Playliste | ID_Player.playlistCollection.item() |
| Playliste mit Namen erzeugen | ID_Player.playlistCollection.newPlaylist() |
| Playliste in die Media-Bibliothek importieren | ID_Player.playlistCollection.importPlaylist() |
| Playliste aus der Media-Bibliothek entfernen | ID_Player.playlistCollection.remove() |
| Playliste aus allen Elemente der Media-Bibliotheks erzeugen | ID_Player.mediaCollection.getAll() |
| Playliste auf einer CD im CD-Laufwerk | ID_Player.cdromCollection.item().playlist |
| Playliste als aktuell setzen | ID_Player.currentPlaylist = zeiger_auf_playliste; |
| aktuelle Playliste | ID_Player.currentPlaylist |
| media Objekt an das Ende der aktuellen Playliste anhängen und damit dort einen Media-Item erzeugen | ID_Player.currentPlaylist.appendItem() |
| Objekt zum CD-Laufwerk ohne CD-Laufwerksbuchstabe: | ID_Player.cdromCollection.item() |
| Objekt zum CD-Laufwerk mit CD-Laufwerksbuchstabe: | ID_Player.cdromCollection.getByDriveSpecifier() |
| aktuell per Player-Control manipulierbares media Objekt bzw. Media Item | ID_Player.controls.currentItem |
| Verfügbarkeit von Controls zu einer Media-Datei / Media-Item | ID_Player.controls.isAvailable() |
| nächsten Playlist-Eintrag (Media Item) vorwärts in der Playliste auswählen | ID_Player.controls.next() |
| Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe | ID_Player.controls.playItem() |
| vorhergehenden Playlist-Eintrag (Media Item) rückwärts in der Playliste auswählen | ID_Player.controls.previous() |
| Anzahl der Wiedergaben (nicht der Wiederholungen) | ID_Player.settings.playCount |
| Wiedergabe-Modus (zufall, endlos) | ID_Player.settings.setMode() |
| Lautstärke für Ton (überschreibt Windows-Lautstärke-Regelung zum Mediumtyp) | ID_Player.settings.volume |

Syntax:

ID_Player.settings.eigenschaft
ID_Player.settings.methode()

Eigenschaften:

| | |
|---------------------|---|
| .autoStart | Autostart der Wiedergabe
wenn Autostart erlaubt:
Wiedergabe sofort gestartet bei Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
wenn Autostart nicht erlaubt ist:
Wiedergabe nach Zuweisung von Werten zu
ID_Player.URL
ID_Player.currentMedia
durch Aufruf von ID_Player.controls.play()
Hinweis: Zuweisung zu ID_Player.URL
ID_Player.currentMedia
setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht |
| .balance | Stereo-Balance (Kanalausrichtung)
-100 bis 100
Standard ist 0
-100 ganz links
100 ganz rechts
0 mittig |
| .baseURL | Basis-Pfad einer HTTP-Url
wird ignoriert von der Url-Angabe im Event ID_Player.scriptCommand (scType mit Wert "URL")
relativer Pfad:
"/" kodierbar, wenn nicht erstes Zeichen
erstes Zeichen darf nicht sein ". "
" "
" \"
" / "
wenn ja, so wird es automatisch gelöscht
".." etc. nicht kodierbar |
| .defaultFrame | Standardname des Frame für Event ID_Player.scriptCommand, wenn scType mit Wert "URL", falls im Kommando-Wert kein anderer Frame kodiert wurde |
| .enableErrorDialogs | automatische Fehleranzeige ein/ aus (true/false) |
| .invokeURLs | Wirksamkeit der Url-Zuweisung an den Standard-Webbrowser aufgrund eines eintreffenden Events
ID_Player.scriptCommand mit scType auf "URL" und Param mit Wert als Kette mit der zuzuweisenden Url
Scriptkommando liegt z.B. in einer ASF-Datei
Achtung: Wenn Script zum Event kodiert wurde, so wird dieses Script vor der Url-Zuweisung abgearbeitet. Das Script ist frei programmierbar und kann z.B. eine Wertänderung von Param und oder ID_Player.settings.invokeURLs durchführen.
Danach wird erst die Url laut Wert von Param an den Webbrowser weitergeleitet, falls das ID_Player.settings.invokeURLs zulässt.
Es muss kein Script zum Event ID_Player.scriptCommand zu scType mit Wert "URL" kodiert sein.
true für weiterreichen
false für nicht weiterreichen |



| | |
|------------------|--|
| .mute | Stummschaltung für Audio bzw. Video mit Ton (true für stumm, false für nicht stumm) |
| .playCount | Anzahl der Wiedergaben (nicht der Wiederholungen) ab 1 |
| .rate | Faktor für Wiedergaberate laut Media-Datei
Veränderung der Wiedergabe:
reale Wiedergaberate = Wiedergaberate laut Medium * Faktor
Floating point
immer 1 bei Audio
< 0 und > 0 bei Medium mit Datenstrom bei ASF oder WMV
< 0 für Rückwärtswiedergabe
> 0 für Vorwärtswiedergabe
> 0 und bei Medium, das nicht Audio und nicht ASF und nicht WMV
< 1 so geringere Rate als laut Medium
> 1 so höhere Rate als laut Medium
Standard ist 1.0 (Rate laut Medium)
aber bei ID_Player.controls.fastForward 5.0
ID_Player.controls.fastReverse -5.0 |
| .volume | Lautstärke für Ton
Achtung: Verändert den Regler zur Media-Datei-Art in der Windows-Lautstärke-Regelung und nicht den Master-Regler !
Bps.: Wiedergabe einer MID-Datei und Volume-Veränderung, so
Regler der MID-Datei verändert
Unbedingt pro Mediumart vor der Volume-Veränderung den aktuellen Wert von
Volume per Script sichern und nach der Wiedergabe rückspeichern !
Integer
0 bis 100
0 für stumm
100 für maximal
Standard laut aktueller Windows Lautstärke-Regelung zum Medium-Typ |
| Methoden: | |
| .getMode() | Modus der Wiedergabe von Tracks (Modus des Players) ermitteln (zufällig, endlos)
siehe Event ID_Player.modeChange |
| .isAvailable() | Veränderbarkeit von ID_Player.settings.rate prüfen |
| .setMode() | Modus der Wiedergabe von Tracks (Modus des Players) setzen (zufällig, endlos)
siehe Event ID_Player.modeChange |

5.3. Webspeech von Logox im Internet Explorer und Netscape

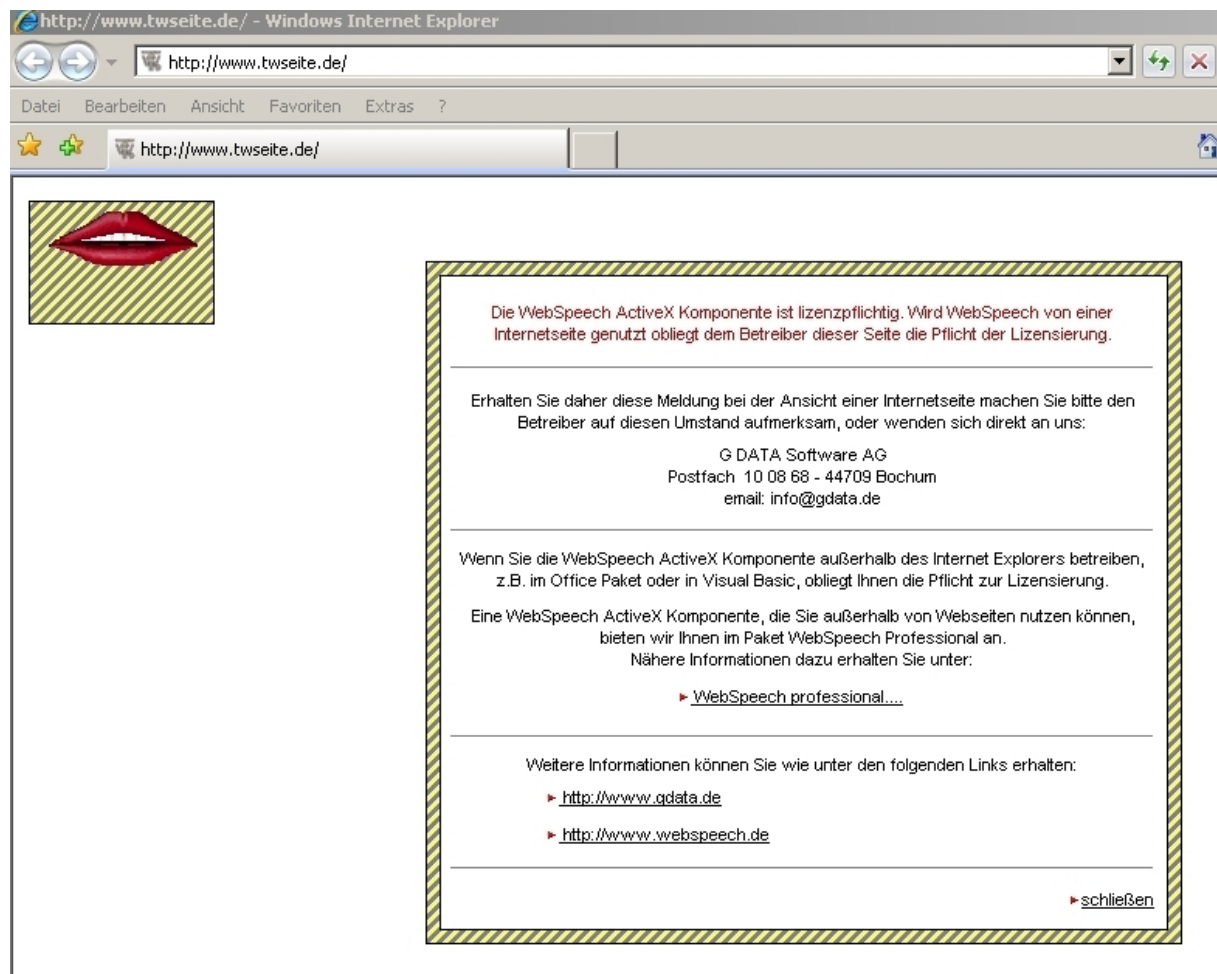
Die Sprachausgabe in einer Webseite ist mit der Software Logox WebSpeech auf Basis der Windows Speech Technology des Unternehmens GDATA möglich (www.gdata.de oder www.logox.de oder www.webspeech.de). Diese Software ist z.T. Freeware und vorrangig auf den Internet Explorer spezialisiert. Es können sprachanimierte Webseiten z..B. behinderten-gerecht erzeugt werden.

Es gibt verschiedene Versionen von Webspeech, die mit dem jeweiligen Plugin bzw. ActiveX.-Control im Browser installiert sein muss. Der Hersteller bietet das Plugin bzw. Active-X-Control für die aktuelle Version von Webspeech an. Ab IE 6.x wird kein Plugin mehr unterstützt. Es muss das Active-X-Control installiert werden. Höhere Versionen sind bezüglich der Steuertags z.B. für Sprachsteuerung nicht kompatibel.

Ab Version 4 wird die Benutzung von Webspeech in einer HTML-Seite auf einem Server (Domain) lizenzpflichtig, wobei die Kosten einer Lizenz

(pro Domain 1 Lizenz) beim Hersteller zu erfragen sind und eventuell mit dem Kauf der Software Logox abgedeckt sind, wenn die Software beim Hersteller registriert wurde und danach auf User-Anfrage nach Ausstellung eines Lizenzschlüssels der Hersteller für die in der Anfrage genannte Domain einen Schlüssel erzeugt hat, der den Domain-Namen enthält. **Für gewerbliche Zwecke sieht der Hersteller andere Regelungen vor, die beim Hersteller zu erfragen sind.** Nur mit der Domainlizenz sind Daten einer Webseite im Internet (auf Webserver) durch den Besucher der Webseite mit seinem Computer (Client) per Active-X-Control des Internet Explorers als Sprache hörbar. (Für lokale Webseiten ohne HTTP-Server bzw. ohne Virtual Host eines lokalen HTTP-Server ist die Domain-Lizenz unerheblich.) Ist der Lizenzschlüssel im HTML-Script bzw. JScript nicht kodiert aber die Webseite auf HTTP-Server gehostet, dann erscheint eine Fehlermeldung - hier ein Beispiel für die Domain www.twseite.de unter dem IE 7:





Wie diese Abfrage funktioniert, konnte der technische Telefon-Support (ca 5 Cent pro Minute) nicht sagen, außer, dass die Schlüssel keine Dongel für Zwangskontakt zu Servern von GData sind. Da aber ein HTTP-Server Routinen des Internetzuganges aktiviert, könnte Webspeech nach Registrierung dieses Status eine per Schlüssel im Scriptcode hinterlegte Zeichenkette per Prüfsumme verglichen mit zulässigen Werten, wobei das Active-X-Control diesen Algorithmus kennen muss. (Es ist auch so, dass bereits Webspeech 2 einen Parameter hatte für einen Schlüssel hatte, der aber nicht benutzbar war.) - Schlaumeier denken jetzt, dass dann Webspeech 2 diese Domain-Zwangslizenz nicht kennt: Korrekt, aber Webspeech 2 läuft nicht unter Windows XP (Tonerzeugung versagt, es liegt nicht am Active-X-Control selbst).

Fatal ist übrigens:

Diese Meldung erscheint auch, wenn der kodierte Domain-Schlüssel zwar von GData gekauft aber vom Active-X-Control nicht akzeptiert wird.

Es ist also nach Erwerb des Domain-Schlüssels zu prüfen, ob dieser akzeptiert wird. Für diese Prüfung muss die Test-Webseite auf einem HTTP-Server gehostet sein. Die Testwebseite index.html hat folgenden Inhalt:

```
<HTML>
<HEAD>
<script language="JavaScript">
<!--
// Browser und WebSpeech Plugin abfragen
//WEBSPEECH FOR NAV DETECTION (MUST BE 1ST)
var IsWebSpeech = 0;
var IsNavigator = 0;
var IsExplorer = 0;
    if(navigator.appName=="Netscape" && navigator.plugins && navigator.javaEnabled)
    {
        IsNavigator=1;

        for(i=0;i<navigator.plugins.length;i++)
        {
            if(navigator.plugins[i].description.indexOf("WebSpeech") != -1 )
            {IsWebSpeech=1;}
        }
    }
}
```



```
// -->
</script>
<!-- WEBSPEECH FOR IE DETECTION (MUST BE 2ND)-->
<script language="VBScript">
on error resume next
if(navigator.javaEnabled) then
IsExplorer=1
IsWebSpeech=IsObject(CreateObject("WebSpeech.WebSpeech"))
end if
</script>
</HEAD>
<BODY>
<script language=Javascript>
// dieser Zweig bestimmt, was WebSpeech tun soll, wenn es installiert ist.
if(IsWebSpeech==1)
{
document.write("<OBJECT ID='WebSpeech1'\n");
document.write("CLASSID=CLSID:B38FEBBF-B2FD-11D3-BEC1-00500445FAEC\n");
document.write("WIDTH=120 HEIGHT=80>\n");
document.write("<PARAM NAME='OPAQUE' VALUE='0'>\n");
document.write("<PARAM NAME='AUTHKEY' VALUE=' www.xxxx.yy @aaaaa-bbbbbb-cccc-ddddd '>\n");
// Domainschlüssel .www.xxxx.yy @aaaaa-bbbbbb-cccc-ddddd
document.write("</OBJECT>\n");
}
else
{
// hier geben Sie an, was passieren soll, wenn WebSpeech nicht installiert ist
document.write("<a href='http://www.webspeech.de/download.php3'> ");
document.write("<img src='pix/schweigen.gif' width='120' height='80' ");
document.write("border='0' alt='Bitte laden Sie WebSpeech!!'></a>\n");
}

if(IsWebSpeech==1)
{alert(window.document.WebSpeech1!=null);
window.document.WebSpeech1.SetText('Ich bin bereit. ');
window.document.WebSpeech1.StartSpeakingImmediate();
}
</Script>
</BODY>
</HTML>
```

Der Vertrieb und Support wurde allerdings von GData eingestellt: Das Produkt ist nicht mehr kaufbar. Selbst wenn man es bei Ebay kauft, muss für den Internet-Einsatz eine Domain-Lizenz beschafft werden. Die Webseiten zu der Sprachausgabe offerieren immer noch das Produkt, ohne dass es je in der Produktübersicht zu finden, geschweige im Onlineshop zu finden ist. Das WebSpeech-Forum ist abgeschaltet (toter Link - Webseite nicht gefunden). Emailsupport hüllt sich auch in Ignoranz. - Die Webseiten sind nichts anderes als Vergraulen von Kunden bzw. Interessenten, dafür bieten sie noch den Download der Free-Versionen und freien Dokumentationen

an, welche für Webseiten, die nicht HTTP-Server gebunden werden, nutzbar sind. Unangenehm ist der Umstand, dass GData die Zwangs-Domain-Lizenz nicht abgeschafft hat, wenn GData das Produkt nicht mehr vertreibt und supportet: Da nur GData die Lizenz vergeben konnte, wird es kritisch bei Problemen mit dem Domainschlüssel, so dass der Kauf der Sprachausgabesoftware nicht

lohnt bei Ebay etc.. Das wird allerdings dafür sorgen, dass das Internet langsamer lebendig wird: Eine Alternative zu Webspeech ist z.B. Microsoft Text To Speech, deren Software unter Win XP bereits als Grundkomponente enthalten ist, aber - im Gegensatz zu Webspeech - nicht mit Soundblaster-kompatiblen Soundkarten (Creative Labs-Standard) funktioniert: Z.B. nicht mit der Soundkarte

Creative Labs Xtreme (Modell SB 04060 mit EAX ab Version 4).

Nachfolgende Beschreibung der Programmierung zu Webspeech in HTML und Script (Javascript und VBScript) gilt ausschliesslich für den privaten und nicht kommerziellen Gebrauch.

Die Programmierung der Versionen 2 bis unter 4 und 4 wird nachfolgend skizziert. Die wenigen Unterschiede von Webspeech 4 zu Vorgänger-Versionen werden explizit genannt.

Bei Nutzung von Webspeech auf einer Domain ist eine Verlinkung auf den Hersteller eventuell notwendig (Hersteller fragen für die Form der Verlinkung) und zwar nicht nur für den Download des Plugins bzw. Active-X-Controls zum Browsers.

Das Software-Paket Logox 4 Professional enthält alle notwendigen Regelungen zur Lizenzierung und Art der Verlinkung und bietet weiteren Support, der eventuell über den Rahmen des kostenlosen Online-Supports des Herstellers hinausgeht.

Für gewerbliche Zwecke sieht der Hersteller andere Regelungen vor, die beim Hersteller zu erfragen sind.

5.3.1. Webspeech-Objekt erzeugen

Unbedingt obige Hinweise zum Domain-Schlüssel beachten !



Wegen Patentwahrung hat Microsoft ein freiwilliges Patch herausgegeben, dass bei ActiveX-Control

per APPLLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.

Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.

Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.

Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.

Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).

DOM der Webseite wird nicht verändert: Nur Blockierung der gesamten Eventsteuerung.

Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.

Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per

fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' WIRD IGNORIERT:

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert: Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizensierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
klick ins Fenster per aktivem Popup



Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende

Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren

oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster

einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer

anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen

Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell

window.focus();

window.document.focus();

if(document.body!=null)

{if(document.body.style!='hidden') // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)

{document.body.focus();}

}

// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt

popupzeiger.show(...);

Hinweis: Der Populfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7



windows.focus() document.focus() und body.focus() funktionieren NICHT
zwischen Register in einem IE-Fenster
zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
.setActive() ist Teilmenge von .focus(): nur das aktiv setzen
funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:
Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:
Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•
http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)



Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}
Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes
• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer



Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements



wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>
(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

5.3.1.1. Webspeech-Objekt in HTML erzeugen

Erzeugung in HTML:

beim IE	anhand des OBJECT-Tags
beim pluginfähigen NS	anhand des EMBED-Tags

zu OBJECT und EMBED:

es sind kodierbar

die üblichen Attribute wie WIDTH, HEIGHT, STYLE etc...

Empfehlung: Angaben zu WIDTH und HEIGHT immer kodieren

Eventhandler, die aber in **keinem Zusammenhang** mit den Events von Webspeech stehen, da diese per VBScript zu kodieren sind hier nicht beschrieben werden.

Beispiel:

```
<OBJECT ID="freies_id"
CLASSID="CLSID:B38FEBBF-B2FD-11D3-BEC1-00500445FAEC"
WIDTH="150"
HEIGHT="125"
> <!-- freies_id muss identisch sein mit der in der EMBED-Kodierung Attribut NAME /-->
```

<!-- nur beim IE //-->

```
<PARAM NAME="AUTHKEY" VALUE="lizenz schluessel">
```

<PARAM NAME="TEXT" VALUE="Ein netter Text">

```
<!-- oder // -->
```

```
<PARAM NAME="URL" VALUE=" http://www.test.de /test.txt">
```

<PARAM NAME="AUTOSTART" VALUE="0">

<PARAM NAME="IMMEDIATE" VALUE="0">

<PARAM NAME="MOUTHANIMATION" VALUE="1">

<PARAM NAME="MOUTHCOLOR" VALUE="black,TEXT,TEXT,TEXT,TEXT">

```
<PARAM NAME="CONTROLPOSITION" VALUE="75">
```

```
<PARAM NAME="TEXTANIMATION" VALUE="3">
```

<PARAM NAME="TEXTCOLOR">VALUE="black.silver">

<PARAM NAME="TEXTPOSITION" VALUE="0">

<PARAM NAME="TEXTSIZE" VALUE="12"

```
<PARAM NAME="TEXTSIZE" VALUE="12">
<PARAM NAME="BACKGROUND_COLOR" VALUE="black">
```

<PARAM NAME="OPAQUE" VALUE="0">

```
<!-- nur beim NS //-->
```

```
<!-- freies id muss identisch sein mit der in der OBJECT-Kodierung Attribut ID -->
```

<!-- PARAM TEXT und PARAM-URL sind alternativ zu kodieren -->

```
<!-- PARAM_TEXT and PARAM_URL sind  
<EMBED NAME="freies_id"  
      TYPE="application/x-WebSpeech"  
      WIDTH=150  
      HEIGHT=125
```



```

PARAM_AUTHKEY="lizenz_schlüssel"
PARAM_TEXT="Ein netter Text."
PARAM_URL="http://www.test.de /test.txt"
PARAM_AUTOSTART="0"
PARAM_IMMEDIATE="0"
PARAM_MOUTHANIMATION="1"
PARAM_MOUTHCOLOR="black,TEXT,TEXT,TEXT,TEXT"
PARAM_CONTROLPOSITION="75"
PARAM_TEXTANIMATION="3"
PARAM_TEXTCOLOR="black,silver"
PARAM_TEXTPOSITION="0"
PARAM_TEXTSIZE="12"
PARAM_BACKGROUNDCOLOR="black"
PARAM_OPAQUE="0"
>
</EMBED>
</OBJECT>

```

Parameter mit ungültigem Wert müssen keinen Fehler erzeugen.

5.3.1.1.1. Parameter AUTHKEY (ab Webspeech 4)

Domain-Lizenz-Schlüssel

kann nicht entfallen, da kein Standard

Solange die Webseite lokal auf dem PC getestet wird, ist keine Lizenz nötig und man kann daher den Schlüssel des Herstellers verwenden:

```

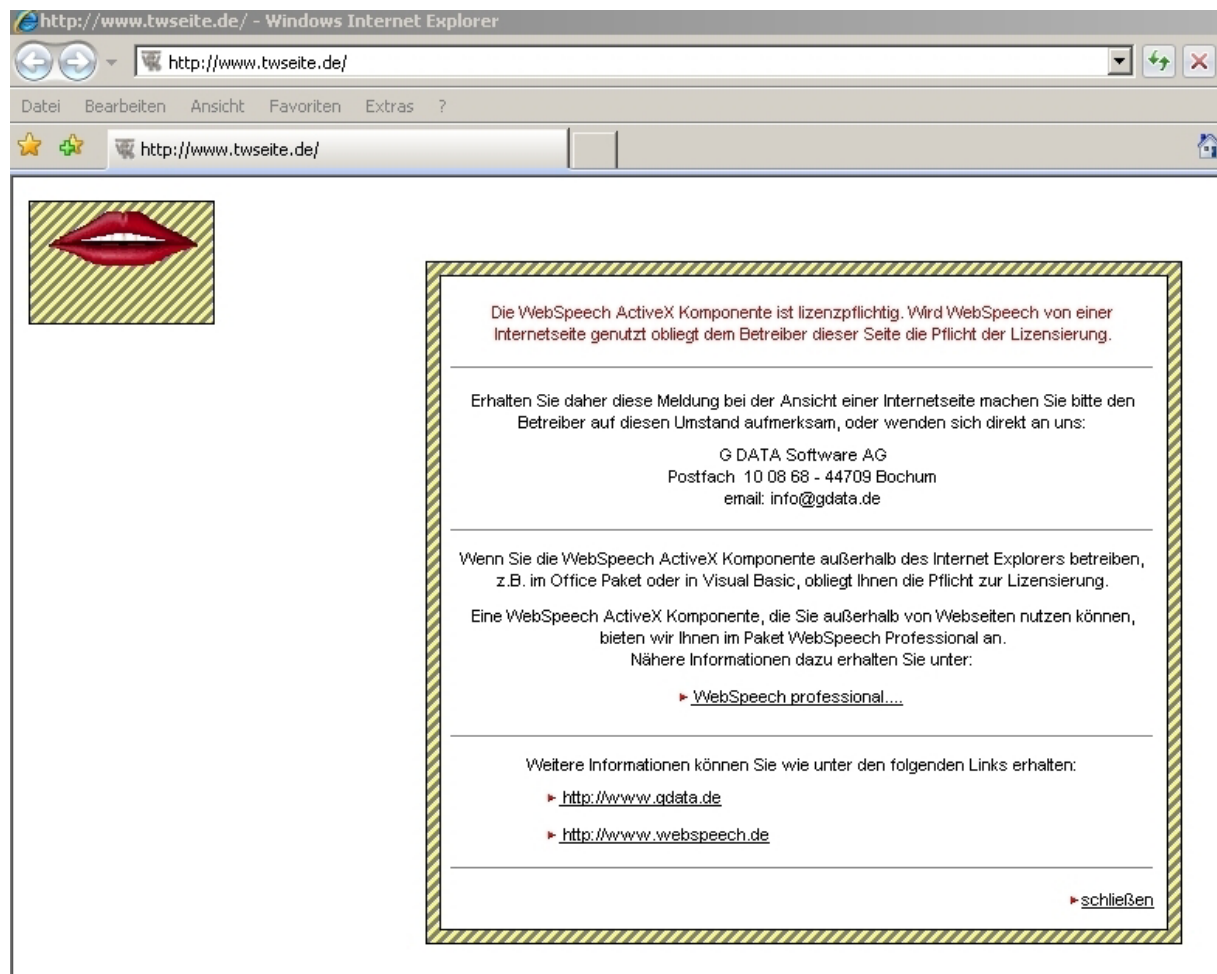
IE      <PARAM NAME="AUTHKEY" VALUE="www.webspeech.de@RGVUA-CSN52-22B98-KU3WD">
NS      PARAM_AUTHKEY="www.webspeech.de@RGVUA-CSN52-22B98-KU3WD"

```

Ab Version 4 wird die Benutzung von Webspeech in einer HTML-Seite auf einem Server (Domain) lizenzpflichtig, wobei die Kosten einer Lizenz

(pro Domain 1 Lizenz) beim Hersteller zu erfragen sind und eventuell mit dem Kauf der Software Logox abgedeckt sind, wenn die Software beim Hersteller registriert wurde und danach auf User-Anfrage nach Ausstellung eines Lizenzschlüssels der Hersteller für die in der Anfrage genannte Domain einen Schlüssel erzeugt hat, der den Domain-Namen enthält. **Für gewerbliche Zwecke sieht der Hersteller andere Regelungen vor, die beim Hersteller zu erfragen sind.** Nur mit der Domainlizenz sind Daten einer Webseite im Internet (auf Webserver) durch den Besucher der Webseite mit seinem Computer (Client) per Active-X-Control des Internet Explorers als Sprache hörbar. (Für lokale Webseiten ohne HTTP-Server bzw. ohne Virtual Host eines lokalen HTTP-Server ist die Domain-Lizenz unerheblich.) Ist der Lizenzschlüssel im HTML-Script bzw. JScript nicht kodiert aber die Webseite auf HTTP-Server gehostet, dann erscheint eine Fehlermeldung - hier ein Beispiel für die Domain www.twseite.de unter dem IE 7:





Wie diese Abfrage funktioniert, konnte der technische Telefon-Support (ca 5 Cent pro Minute) nicht sagen, außer, dass die Schlüssel keine Dongel für Zwangskontakt zu Servern von GData sind. Da aber ein HTTP-Server Routinen des Internetzuganges aktiviert, könnte Webspeech nach Registrierung dieses Status eine per Schlüssel im Scriptcode hinterlegte Zeichenkette per Prüfsumme vergleichen mit zulässigen Werten, wobei das Active-X-Control diesen Algorithmus kennen muss. (Es ist auch so, dass bereits Webspeech 2 einen Parameter hatte für einen Schlüssel hatte, der aber nicht benutzbar war.) - Schlaumeier denken jetzt, dass dann Webspeech 2 diese Domain-Zwangslizenz nicht kennt: Korrekt, aber Webspeech 2 läuft nicht unter Windows XP (Tonerzeugung versagt, es liegt nicht am Active-X-Control selbst).

Fatal ist übrigens:

Diese Meldung erscheint auch, wenn der kodierte Domain-Schlüssel zwar von GData gekauft aber vom Active-X-Control nicht akzeptiert wird.

Es ist also nach Erwerb des Domain-Schlüssels zu prüfen, ob dieser akzeptiert wird. Für diese Prüfung muss die Test-Webseite auf einem HTTP-Server gehostet sein. Die Testwebseite index.html hat folgenden Inhalt:

```
<HTML>
<HEAD>
<script language="JavaScript">
<!--
// Browser und WebSpeech Plugin abfragen
//WEBSPEECH FOR NAV DETECTION (MUST BE 1ST)
var IsWebSpeech = 0;
var IsNavigator = 0;
var IsExplorer = 0;
    if(navigator.appName=="Netscape" && navigator.plugins && navigator.javaEnabled)
    {
        IsNavigator=1;

        for(i=0;i<navigator.plugins.length;i++)
        {
            if(navigator.plugins[i].description.indexOf("WebSpeech") != -1 )
            {IsWebSpeech=1;}
        }
    }
}
```



```
// -->
</script>
<!-- WEBSPEECH FOR IE DETECTION (MUST BE 2ND)-->
<script language="VBScript">
on error resume next
if(navigator.javaEnabled) then
IsExplorer=1
IsWebSpeech=IsObject(CreateObject("WebSpeech.WebSpeech"))
end if
</script>
</HEAD>
<BODY>
<script language=Javascript>
// dieser Zweig bestimmt, was WebSpeech tun soll, wenn es installiert ist.
if(IsWebSpeech==1)
{
document.write("<OBJECT ID='WebSpeech1'\n");
document.write("CLASSID=CLSID:B38FEBBF-B2FD-11D3-BEC1-00500445FAEC\n");
document.write("WIDTH=120 HEIGHT=80>\n");
document.write("<PARAM NAME='OPAQUE' VALUE='0'>\n");
document.write("<PARAM NAME='AUTHKEY' VALUE=' www.xxxx.yy @aaaaa-bbbbbb-cccc-ddddd '>\n");
// Domainschlüssel .www.xxxx.yy @aaaaa-bbbbbb-cccc-ddddd
document.write("</OBJECT>\n");
}
else
{
// hier geben Sie an, was passieren soll, wenn WebSpeech nicht installiert ist
document.write("<a href='http://www.webspeech.de/download.php3'> ");
document.write("<img src='pix/schweigen.gif' width='120' height='80' ");
document.write("border='0' alt='Bitte laden Sie WebSpeech!!'></a>\n");
}

if(IsWebSpeech==1)
{alert(window.document.WebSpeech1!=null);
window.document.WebSpeech1.SetText('Ich bin bereit. ');
window.document.WebSpeech1.StartSpeakingImmediate();
}
</Script>
</BODY>
</HTML>
```

Der Vertrieb und Support wurde allerdings von GData eingestellt: Das Produkt ist nicht mehr kaufbar. Selbst wenn man es bei Ebay kauft, muss für den Internet-Einsatz eine Domain-Lizenz beschafft werden. Die Webseiten zu der Sprachausgabe offerieren immer noch das Produkt, ohne dass es je in der Produktübersicht zu finden, geschweige im Onlineshop zu finden ist. Das WebSpeech-Forum ist abgeschaltet (toter Link - Webseite nicht gefunden). Emailsupport hüllt sich auch in Ignoranz. - Die Webseiten sind nichts anderes als Vergraulen von Kunden bzw. Interessenten, dafür bieten sie noch den Download der Free-Versionen und freien Dokumentationen

an, welche für Webseiten, die nicht HTTP-Server gebunden werden, nutzbar sind. Unangenehm ist der Umstand, dass GData die Zwangs-Domain-Lizenz nicht abgeschafft hat, wenn GData das Produkt nicht mehr vertreibt und supportet: Da nur GData die Lizenz vergeben konnte, wird es kritisch bei Problemen mit dem Domainschlüssel, so dass der Kauf der Sprachausgabesoftware nicht

lohnt bei Ebay etc.. Das wird allerdings dafür sorgen, dass das Internet langsamer lebendig wird: Eine Alternative zu Webspeech ist z.B. Microsoft Text To Speech, deren Software unter Win XP bereits als Grundkomponente enthalten ist, aber - im Gegensatz zu Webspeech - nicht mit Soundblaster-kompatiblen Soundkarten (Creative Labs-Standard) funktioniert: Z.B. nicht mit der Soundkarte

Creative Labs Xtreme (Modell SB 04060 mit EAX ab Version 4).

5.3.1.1.2. Parameter TEXT und URL

Text der Sprachausgabe bei Erzeugung des Webspeech-Objektes

kann entfallen, aber kein Standard

es ist entweder TEXT oder URL zu kodieren

TEXT Wert ist String also in " " bzw. ' ' zu kodieren
kann Sprechtags enthalten: Achtung Sprechtags vor Webspeech 4 sind in der Regel nicht kompatibel zu Webspeech 4

Beispiel für Webspeech 2: "Herzlich willkommen bei \PAU=500\ WebSpeech."

Beispiel für Webspeech 4: "Herzlich willkommen bei #PAU=500# WebSpeech."

URL Pfad und Name der Datei

Datei: ASCII-Textdatei
LOGOX-Datei

mit Suffix .TXT
mit Suffix LGX

Datei ist mit Tools zu Webspeech zu erstellen

vor Webspeech 2 reine ASCII-Datei mit Versionsnummer

in



Zeile 1

Datei kann im Internet liegen
lokal liegen

ab Webspeech 4 kompilierte Datei
Beispiel: "http://www.test.de/text.txt"
Beispiel: "file:///C:/Texte/readme.txt"

5.3.1.1.3. Parameter AUTOSTART

:automatische Sprachausgabe mit Vollendung der Erzeugung des Webspeech-Objektes
Parameter TEXT oder URL muss mit gültigem Wert ungleich Leerkette kodiert worden sein

"0" Standard
keine automatische Sprachausgabe: User muss Play-Button des Webspeech-Control klicken
Achtung: User weiss nicht, ob ein Text beim Erzeugen des Webspeech-Objektes erzeugt wurde oder nicht !

1 für Autostart

5.3.1.1.4. Parameter IMMEDIATE

es muss Autostart aktiv sein
Text sofort sprechen und dabei eine eventuell bereits aktive Sprachausgabe beachten

"0" Standard
warten bis bereits aktives Sprechen beendet ist

"1" bereits aktives Sprechen sofort beenden

5.3.1.1.5. Parameter MOUTHANIMATION

"0" keine Mundanimation bei der Sprachausgabe
"1" Standard
Mundanimation bei der Sprachausgabe

5.3.1.1.6. Parameter MOUTHCOLOR

Farbgestaltung der Mundregion

Farben sind HTML-Farben browserspezifisch
"#rrggbb" mit rr für Rotanteil, gg für Grünanteil, bb für Blauanteil
jeweils hexadezimale Kodierung
"DEFAULT" für Standardfarben
"TRANSPARENT" oder "TRANSP" für Transparenz, also Hintergrund nicht abdecken, also Farbe nicht anzeigen
"TEXTURE" oder "TEXT" für eine vordefinierte Textur-Grafik
wenn TEXTURE verwendet, so wird Angabe der Umrandung ignoriert

Mundbereiche sind Umrandung
Rachen
Zunge
Zähne
Lippen

Wert String aus komma-getrennten Farbangaben in folgender Reihenfolge
Farbe der Umrandung
Farbe des Rachens
Farbe der Zunge
Farbe der Zähne
Farbe der Lippen
Komma sind Teil des String

Standard ist "black, text, text, text, text"

5.3.1.1.6. Parameter TEXTANIMATION

Anzeige des Textes während der Sprachausgabe

"0" keine Textanzeige
"1" Lauftext von rechts nach links
"2" wortweise
"3" Standard
laut UserEinstellung

5.3.1.1.7. Parameter TEXTCOLOR

Textfarbe

<PARAM NAME="TE^{TEXTCOLOR}" VALUE="#A1B2C3, forestgreen Text Schatten black ist Standard gray ist Standard
>

Farben sind HTML-Farben browserspezifisch
"#rrggbb" mit rr für Rotanteil, gg für Grünanteil, bb für Blauanteil
jeweils hexadezimale Kodierung
"DEFAULT" für Standardfarben



"TRANSPARENT" oder "TRANSP" für Transparenz, also Hintergrund nicht abdecken, also Farbe nicht anzeigen
 Wert String aus komma-getrennten Farbangaben in folgender Reihenfolge
 Farbe des Textes
 Farbe des Schattens
 Komma sind Teil des String
 Standard ist "black, gray"

5.3.1.1.8. Parameter TEXTPOSITION

vertikale Textposition relativ zur Lage des OBJECT-Tags bzw. EMBED-Tags
 "0" bis "100"
 "0" Standard ganz oben
 "100" ganz unten.

5.3.1.1.9. Parameter TEXTSIZE

Textfonthöhe in Pixel
 ab "0"
 ohne Einheit kodieren
 Standard ist "12"

5.3.1.1.10. Parameter BACKGROUNDCOLOR

Hintergrundfarbe des Webspeech-Objektes innerhalb der Dimension laut OBJECT-Tag bzw. EMBED-Tag (Width und Height), wobei alle sichtbaren Komponenten des Objektes auf der Hintergrundfarbe liegen, also diese abdecken, falls Komponenten nicht transparent sind

Farben sind	HTML-Farben "#rrggbb"	browserspezifisch mit rr für Rotanteil, gg für Grünanteil, bb für Blauanteil jeweils hexadezimale Kodierung für Standardfarbe white
	"DEFAULT"	

Hinweis: Transparenz siehe Parameter OPAQUE

5.3.1.1.11. Parameter OPAQUE

Deckkraft der per Parameter BACKGROUNDCOLOR definierten Hintergrundfarbe
 "0" transparent, also Hintergrundfarbe nicht anzeigen
 "1" deckend, also Hintergrundfarbe anzeigen

5.3.1.1.12. Parameter CONTROLPOSITION

vertikale Control-Elemente-Leiste relativ zur Lage des OBJECT-Tags bzw. EMBED-Tags
 "0" bis "100"
 "0" ganz oben
 "100" ganz unten.
 "75" Standard

5.3.1.2. Webspeech-Objekt in Javascript erzeugen

per Funktion document.write()

5.3.2. Webspeech-Objekt in Javascript verwalten

5.3.2.1. Webspeech-Objekt in Javascript erkennen

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
<!------- WebSpeech Erkennung ----->
function NS_Pruefen()
{
    var Wert = 0;

    // prüfen auf NS mit Plugin-Fähigkeit
    if(
        (navigator.appName=="Netscape")
        && (navigator.plugins)
        && (navigator.javaEnabled)
    )
    {
        // pluginfähiger NS gefunden

        // prüfen auf Webspeech2-Plugin
        var PluginFeld = navigator.plugins
        var AnzahlPlugins = PluginFeld.length;

        for( var i=0; i < AnzahlPlugins; i++)
        {
            var PluginBeschreibung = PluginFeld[i].description;
```



```

        if(PluginBeschreibung.indexOf("WebSpeech.2") != -1 )
        {Wert=1;} // Plugin gefunden
    }
}

return Wert;
}

<!-- Schritt 1: Webspeech-Erkennung für NS mit Pluginfähigkeit //-->
var IsWebSpeech = NS_Pruefen();      // IsWebSpeech muss global sein
</SCRIPT>

<!-- Schritt 2: Webspeech-Erkennung für IE nur per Visual Basic möglich //-->
<SCRIPT LANGUAGE="VBScript">
on error resume next
if(navigator.javaEnabled) then
    IsWebSpeech=IsObject(CreateObject("WebSpeech.WebSpeech"))
end if
</SCRIPT>

<!-- Schritt 3: Webspeech-Erkennung für Opera //-->
<SCRIPT LANGUAGE="JavaScript">
    // Opera kann kein Webspeech (kann kein ActiveX noch NAV-Plugin
    //                               und Opera-Plugin existiert nicht)
if (navigator.userAgent.indexOf("Opera") != -1)
{IsWebSpeech = 0;}
</SCRIPT>

<!------- WebSpeech Routinen ----->

<SCRIPT LANGUAGE="Javascript">

    <!------- vorlesen ----->

    function vorlesen()
    {
        if(IsWebSpeech==1)
        {
            var Version = WebSpeechObjekt.GetVersion();
            {alert(Version); } // z.B. 4. 0. 2. 34 bei Webspeech 4

            HauptVersion = Version.slice(0,1);
            var Text = "Webspeech " + HauptVersion + " erkannt."; // z.B. 4 bei Webspeech 4

            WebSpeechObjekt.SetText(Text);
            WebSpeechObjekt.StartSpeaking();
        }
    }

    <!------- WebSpeech Init ----->

    var WebSpeechObjekt;      // muss global sein

    function init()
    {
        if(IsWebSpeech==1)
        {
            document.writeln("<OBJECT    ID='ID_WebSpeech'
                                CLASSID=CLSID:B38FEBBF-B2FD-11D3-BEC1-00500445FAEC'
                                WIDTH=110
                                HEIGHT=100
                                >"); // alles 1 Zeile
            document.writeln("<PARAM    NAME='AUTHKEY'
                                VALUE='www.webspeech.de@RGVUA-CSN52-22B98-KU3WD'
                                >"); // alles 1 Zeile
            document.writeln("<PARAM NAME='AUTOSTART' VALUE='1'>");
            document.write("<PARAM NAME='TEXTSIZE' VALUE='8'>\n");
            document.write("<PARAM NAME='TEXTANIMATION' VALUE='2'>\n");
            document.write("<PARAM NAME='TEXTCOLOR' VALUE='black,transparent'>\n");

            // NS
            document.write("<EMBED\n");
            document.write("    NAME='ID_WebSpeech '
                            TYPE='application/x-WebSpeech' WIDTH='110' HEIGHT='100'\n"

```



```

        ); // alles 1 Zeile
        document.write("PARAM_AUTOSTART='1'\n");
        document.write("PARAM_TEXTSIZE='8'\n");
        document.write("PARAM_TEXTANIMATION='2'\n");
        document.write("PARAM_TEXTCOLOR='black,transparent'\n");
        document.write(">\n");
        document.write("</EMBED>\n");

        document.write("</OBJECT>\n");

        WebSpeechObjekt = window.document.ID_WebSpeech; // NS und IE
    }
    else
    { alert('Es ist kein WebSpeech installiert'); }
}

//-->
</SCRIPT>
</HEAD>
<BODY onLoad="init();vorlesen();">
</BODY>
</HTML>

```

5.3.2.2. *WebSpeech-Objekt in Javascript programmieren*

5.3.2.2.1. *WebSpeech-Objekt und seine Methoden*

Die Methoden lassen die vollständige Verwaltung der WebSpeech-Objektes bezüglich der Eigenschaften der Sprachausgabe zu und das auch nach der Erzeugung des WebSpeech-Objektes.

Parameter aus der HTML-Kodierung des WebSpeech-Objektes sind als Javascript-Methoden entweder direkt oder indirekt ansprechbar. Es ist möglich, in der HTML-Kodierung keine Parameter anzugeben und diese durch Javascript zu ersetzen.

Hinweis: Entgegen den Beschreibungen von Logox liefern Funktionen, die einen Rückkehrcode besitzen sollen, diesen z.T. **nicht**:

Unter der kostenpflichtigen Version des Active-X-Controls von WebSpeech 4 in Verbindung mit Logox 4 **Professional** mit Serverupdate 1 (fehlerfreie Installation),

sowie bei Nutzung des IE 6.0 SP1 in W9x (Netscape wird von WebSpeech 4 nicht mehr unterstützt),

sowie mit Kodierung sämtlicher Scriptanweisungen für WebSpeech (inklusive der Erzeugung des WebSpeech-Objektes) per JScript (nicht JavaScript) im HEAD des HTML-Dokumentes,

wird für die Abfrage des Rückkehrcodes o.g. Funktionen **undefined** geliefert.

Desweiteren wird die Zeigerabfrage auf eine WebSpeech-Funktion per

zeiger_auf_webspeech_objekt.webspeech_funktion != null

vom Browser mit der Meldung quittiert, dass das Objekt die Funktion **nicht unterstützt**, obwohl das Objekt existiert. (Anstelle zeiger_auf ... ist natürlich der konkrete Zeiger und anstelle webspeech_funktion die konkrete zulässige Funktion zu kodieren). Die trotzdem durchgeführte Ausführung der o.g. WebSpeech-Funktionen, die keinen Rückkehrcode liefern, kann man an den **manuellen** Steuerungselementen des WebSpeech-Objektes testen.

Unter JScript in Verbindung mit einer Rückkehrcode-Abfrage nutzbare Funktionen sind IsMuting(), IsSpeaking(), IsPause(), IsLoading(), jedoch nicht z.B. SetText():

In der Kodierung if (zeiger_auf_webspeech_objekt.SetText("Testtext")){...}else{...} liefert if den Wert undefined, so dass die if-Steuerung hinfällig ist.

Die Kodierung zeiger_auf_webspeech_objekt.SetText("Testtext"); führt die Funktion aus (wenn das Objekt existiert), gibt aber keine Auskunft, dass die Funktion ausgeführt wurde (Erfolg oder Misserfolg). SetText() kann also nicht für kontrolliertes Laden eines Textes benutzt werden. Nur das manuelle Klicken auf das Sprech-Symbol in der Steuerungsleiste zum WebSpeech-Objekt gibt Auskunft. Wenn der Test des HTML-Dokumentes mit WebSpeech erfolgreich war, heisst das aber nicht, dass die Funktion SetText() auch immer im Onlinezustand des HTML-Dokumentes funktioniert. Damit hat der Programmierer das Nachsehen.: Die Steuerung des WebSpeech-Objektes ist schlichtweg **unvollständig** implementiert.

Die Kodierung if (zeiger_auf_webspeech_objekt.SetText != null) liefert die Browsermeldung, dass SetText nicht vom Objekt unterstützt wird, obwohl die Kodierung zeiger_auf_webspeech_objekt.SetText("Testtext"); die Funktion ausführt, wenn das Objekt existiert.

Diese Situation ist äußerst bedauerlich ! Aufgrund eigener Erfahrungen des Autors dieser JavaScript-Dokumentation (der in Besitz mehrerer gekaufter Lizenzen von WebSpeech 4 ist) bezüglich des (für deutsche Verhältnisse typischen) Kundenservices des deutschen Logox-Software-Anbieters (schwerfälliger bzw. z.T. unterlassener Email-Service (selbst bei Beantragung eines Autoren-Schlüssels aufgrund gekaufter und registrierter Lizenz), Alternativ-Service auf Basis einer 0190-Telefonnummer mit Kosten in Höhe von 1,86 Euro pro Minute, unzureichende Pflege bzw. dürftiger Inhalt des Internet-Forums) hat der Autor dieser JavaScript-Dokumentation eine Supportanfrage beim o.g. Softwareanbieter unterlassen. Der (weitere bzw. zukünftige) Einsatz von Webspeech will also aus kosten- und programmtechnischen Gründen genau überlegt sein. Wer z.B. den Versuch scheut, o.g. fehlende Rückkehrcode durch eigenen Programmierungsaufwand per JScript zu umgehen, sollte o.g. WebSpeech nicht oder nur unter Duldung o.g. Nachteile einsetzen.

.GetInterfaceVersion() Haupt-Version von Webspeech ermitteln, das auf dem PC des Users installiert ist
Syntax:



.SetBackgroundColor()

entspricht BACKGROUNDCOLOR

Hintergrundfarbe des Webspeech-Objektes einstellen

Deckkraft der Hintergrundfarbe einstellen per .SetOpaque()

Syntax:

[var Wert =] zeiger_auf_webspeech_objekt.SetBackgroundColor(Kette)

Kette	String	
	Farbangabe	
		HTML-Farben
		#RRGGBB
		DEFAULT (Standardfarben),

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags
		NAME des EMBED-Tags

.SetControlPosition()

entspricht Parameter CONTROLPOSITION

vertikale Position der Steuerungselemente für User relativ zur Lage des Webspeech-Objektes

Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt.SetControlPosition(Wert1);

Wert1	Integer	
	0 bis 100	
	0	ganz oben
	100	ganz unten

Wert2	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags
		NAME des EMBED-Tags

.SetKey()

ab Webspeech 4

entspricht Parameter AUTHKEY

Lizenz-Schlüssel festlegen

für lokale Test des HTML-Dokumentes ohne Internetzugriff auf die Domain kann der herstellereigene Schlüssel verwendet werden:

"www.webspeech.de@RGVUA-CSN52-22B98-KU3WD"

Syntax:

[var Wert =] zeiger_auf_webspeech_objekt..SetKey(Kette)

Kette	String
	Author-Schlüssel laut Lizenz

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags
		NAME des EMBED-Tags

.SetMouthAnimation()

entspricht Parameter MOUTHANIMATION

Mundanimation einstellen

Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt..SetMouthAnimation(Wert1);

Wert1	Integer	
	1	Mund wird mitbewegt
	0	Mund wird nicht mitbewegt

Wert2	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags
		NAME des EMBED-Tags

.SetMouthColor()

entspricht Parameter MOUTHCOLOR

Mundfarben einstellen



Syntax:

[var Wert =] zeiger_auf_webspeech_objekt.SetMouthColor(Kette)

Kette Liste von kommagetrennten Farbangaben
als String (inklusive Kommas)
Listenelemente mit nachfolgender Reihenfolge
Farbe der Umrandung
Farbe des Rachens
Farbe der Zunge
Farbe der Zähne
Farbe der Lippen

HTML-Farben
#RRGGBB
DEFAULT (Standardfarben),
TRANSPARENT oder TRANSP (Hintergrundtransparenz)
TEXTURE oder TEXT Grafik in die aktuelle Form des Mundes
wenn TEXTURE verwendet, so wird Angabe der Umrandung ignoriert

Wert Integer
 >= 0 so erfolgreich
 < 0 so nicht erfolgreich

zeiger_auf_webspeech_objekt laut ID des OBJECT-Tags
NAME des EMBED-Tags

.SetOpaque()

entspricht Parameter OPAQUE
Deckkraft der Hintergrundfarbe des Webspeech-Objektes einstellen
Hintergrundfarbe definiert per .SetBackgroundColor()
Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt.SetOpaque(Wert1);

Wert1 Integer
 0 transparent, also Hintergrundfarbe des Webspeech-Objektes nicht sichtbar
 dafür die Hintergrundfarbe des Containers vom Webspeech-Objekt
 1 Hintergrundfarbe deckt ab, also nur Hintergrundfarbe des
 vom Webspeech-Objekt sichtbar und nicht die des Containers

Wert2 Integer
 >= 0 so erfolgreich
 < 0 so nicht erfolgreich

zeiger_auf_webspeech_objekt laut ID des OBJECT-Tags
NAME des EMBED-Tags

.SetText()

entspricht Parameter TEXT
zu sprechenden Text puffern für die **nächste** Sprachausgabe (Sprechtags möglich im Text)
bereits gesprochener Text wird nicht beeinflusst
Sprachausgabe starten mit StartSpeaking()
siehe .LoadText()
Syntax:

[var Wert =] zeiger_auf_webspeech_objekt.SetText(Kette);

Kette String, auch Literal
kann Sprechtags enthalten: Achtung Sprechtags vor Webspeech 4 sind in
der Regel nicht kompatibel zu Webspeech 4

Beispiel für Webspeech 2:
"Herzlich willkommen bei \pau=500\ WebSpeech."

Beispiel für Webspeech 4:
"Herzlich willkommen bei #PAU=500# WebSpeech."

Wert Integer
 >= 0 so erfolgreich
 < 0 so nicht erfolgreich

zeiger_auf_webspeech_objekt laut ID des OBJECT-Tags
NAME des EMBED-Tags

.SetTextAnimation()

entspricht Parameter TEXTANIMATION
Anzeige des Textes während der Sprachausgabe einstellen
Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt..SetTextAnimation(Wert1)



Wert1	Integer 0 bis 3	
	0	keine Anzeige
	1	als Lauftext von rechts nach links
	2	einzelne Wörter erscheinen nacheinander
	3	laut Usereinstellung

Wert2	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags NAME des EMBED-Tags
-----------------------------	------	---

.SetTextColor()entspricht Parameter TEXTCOLOR
Textfarben einstellen

Syntax:

[var Wert =] zeiger_auf_webspeech_objekt.SetTextColor(Kette)

Kette	Liste	von kommasetrennten Farbangaben als String (inklusive Kommas) Listenelemente mit nachfolgender Reihenfolge Farbe des Textes Farbe des Schattens
		HTML-Farben #RRGGBB DEFAULT (Standardfarben), TRANSPARENT oder TRANSP (Hintergrundtransparenz)

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags NAME des EMBED-Tags
-----------------------------	------	---

.SetTextPosition()

entspricht Parameter TEXTPOSITION

vertikale Position der Textanzeige während der Sprachausgabe relativ zur Lage des Webspeech-Ojektes

Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt.SetTextPosition(Wert1);

Wert1	Integer 0 bis 100	
	0	ganz oben
	100	ganz unten

Wert2	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags NAME des EMBED-Tags
-----------------------------	------	---

.SetTextSize()

entspricht Parameter TEXTSIZE

Fonthöhe der Textanzeige während der Sprachausgabe

Syntax:

[var Wert2 =] zeiger_auf_webspeech_objekt.SetTextSize(Wert1);

Wert1	Integer, in Pixel
	>0

Wert2	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID des OBJECT-Tags NAME des EMBED-Tags
-----------------------------	------	---

.StartSpeaking()

nächste Sprachausgabe starten mit Abwarten auf das Ende einer bereits aktiven Sprachausgabe

wiederholbar ohne erneute Pufferung von Text

es muss ein Text gepuffert sein (siehe SetText() und LoadText()), sonst erfolgt keine Sprachausgabe

siehe .StopSpeaking() und .PauseSpeaking()

Syntax:



```
[ var Wert = ] zeiger_auf_webspeech_objekt.StartSpeaking();
```

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID	des OBJECT-Tags
		NAME	des EMBED-Tags

.StartSpeakingImmediate() nächste Sprachausgabe starten ohne Abwarten auf das Ende einer bereits aktiven Sprachausgabe, die sofort beendet wird (auch wenn diese pausiert)
wiederholbar ohne erneute Pufferung von Text
es muss ein Text gepuffert sein (siehe SetText() und LoadText()), sonst erfolgt keine Sprachausgabe
siehe .StopSpeaking() und .PauseSpeaking()
Syntax:

```
[ var Wert = ] zeiger_auf_webspeech_objekt.StartSpeaking();
```

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID	des OBJECT-Tags
		NAME	des EMBED-Tags

.StopSpeaking() aktive Sprachausgabe beenden (auch wenn diese pausiert)
siehe .StartSpeaking() und .StartSpeakingImmediate()
Syntax:

```
[ var Wert = ] zeiger_auf_webspeech_objekt.StopSpeaking();
```

Wert	Integer	
	>= 0	so erfolgreich
	< 0	so nicht erfolgreich

zeiger_auf_webspeech_objekt	laut	ID	des OBJECT-Tags
		NAME	des EMBED-Tags

5.3.2.2.2. Webspeech-Objekt und Events

Events in Javascript existieren nicht. Alternativ sind die Methoden isXXX() zu verwenden
Events sind nur über VBScript realisierbar .

5.3.2.2.3. Beispiel zur Javascript-Programmierung zu Webspeech 2 für IE und NS 4.x

Hinweis: Die Programmierung für die WebSpeech-Version höher als 2 ist
nur z.T. abwärtskompatibel zur WebSpeech-Version 2
nicht im selben Umfang möglich wie wie WebSpeech-Version 2 (siehe obige Bemerkungen)

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    // ##### Webspeech-Prüfung auf ActiveX des IE
    var IE_ActivexInstalliert=false;           // nur IE: Annahme: ActiveX ist nicht installiert

//-->
</SCRIPT>

<SCRIPT LANGUAGE="VBScript">
on error resume next

if (navigator.javaEnabled) then
IE_ActivexInstalliert = IsObject(CreateObject("WebSpeech.WebSpeech.2"))
end if
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript1.2">
<!--
function WebSpeechInstallationPruefen()
{
    var ReturnWert=0;

    // 0 für Webspeech-Plugin ist nicht installiert
    //    im Browser (egal ob NS oder IE)
    //    oder IE bzw. NS nicht aktiv
    // >0 für Webspeech-Plugin ist installiert im NS
    //    und NS aktiv
    // <0 für Webspeech-Plugin ist installiert im IE
    //    und IE aktiv

    if (ns)
```



```

    {
        if ( (navigator.plugins)           // Plugins-Objekt existiert
            && (navigator.javaEnabled)     // Java ist aktiv im Browser
        )
        {
            // prüfen auf eingebundenes Plugin "Webspeech.2"
            // Objekt navigator.plugins[] ist Feld
            // mit Anzahl der installierten Plugins lauf navigator.plugins.lenght
            // ist leer, wenn kein Plugin (egal welcher Hersteller) installiert ist
            // im Browser, also Anzahl = 0
            // Achtung: Feld-Index immer ab 0 !!

            var Zahler=0;                    // Index ab 0
            while ( (Zahler < navigator.plugins.lenght) // Länge ab 1
                && (ReturnWert = 0)           // laut Deklaration
            )
            {
                if (navigator.plugins[Zahler].description.indexOf("WebSpeech.2") != -1)
                {ReturnWert=1;}              // Webspeech-Plugin ist installiert

                Zahler++;
            }
        }
    }
else
{
    if (ie)
    {
        if (IE_ActivexInstalliert)         // siehe oben
        {
            // IE aktiv und ActiveX-Control installiert
            ReturnWert=-1;
        }
    }
}

return ReturnWert;
}

// ++++++ Routinen zur WebSpeech-Ausgabe
function WebSpeech_TextStringSprechen (Text, Warten)
// benötigt erzeugtes Standard-WebSpeech-Objekt
// Text : nur String, Kann leer sein,
// also nur zum LÖSCHEN des Objekt-Text-Puffers, damit Useraktion
// auf Control-Leiste keine Ausgabe auslöst
// Warten : 0 Sprachausgabe sofort starten und bereits aktive abbrechen
// 1 Sprachausgabe erst starten, wenn aktive beendet ist
{
    if (WebSpeechInstalliert != 0)
    {
        parent.info.WebSpeech_TextTemporaerLadenUndSprechen( WebSpeechStandardObjektNr,
                                                                0,
                                                                Text,
                                                                Warten
                                                                );
    }
}

// ++++++ auf erzeugtes Objekt prüfen
function WebSpeech_AufErzeugtesHTMLObjektPruefen(ObjektNr)
// liefert true, wenn Objekt erzeugt ist
// sonst false
// ObjektNr muss korrekt sein und wird nicht geprüft !
// nur ein erzeugtes Objekt ist auch sichtbar
{
    return (WebSpeechObjektZeigerFeld[ObjektNr] != null);
}

// ++++++ Zustand eines erzeugtes WebSpeech-Objekt ermitteln
// benötigt WebSpeechInstalliert
function WebSpeech_HTMLObjektZustandErmitteln(ObjektNr)
// aktualisiert WebSpeechObjektZustandFeld[]
// liefert -1 Fehler weil Objektnummer falsch und oder Objekt nicht installiert
// 0 unbekannter Zustand

```



```

//      1      Ausgabe gerade begonnen worden
//      2      gestartete Ausgabe komplett beendet
//      3      gestartete Ausgabe pausiert
//      4      gestoppte Ausgabe gerade erneut gestartet
//      5      Ausgabe ist vom User stummgeschaltet (muted)
//      6      Text wird gerade geladen
{
    var ReturnWert=-1;          // Annahme: Objekt nicht installiert oder ObjektNr ist falsch

    if (WebSpeechObjektZeigerFeld[ObjektNr])
    {
        //      erst Ereignisse abfragen
        ReturnWert=WebSpeechObjektZustandFeld[ObjektNr];

        if (ReturnWert == 0)
        {
            // unbekannter Zustand also Is-Routinen bemühen
            //      auf Muten prüfen
            if (WebSpeechObjektZeigerFeld[ObjektNr].IsMuting())
            {
                ReturnWert=5;

                // und WebSpeechObjektZustandFeld korrigieren
                WebSpeechObjektZustandFeld[ObjektNr]=5;
            }
            else
            {
                // auf gerade sprechen prüfen
                if (WebSpeechObjektZeigerFeld[ObjektNr].IsSpeaking())
                {
                    ReturnWert=1;

                    // und WebSpeechObjektZustandFeld korrigieren
                    WebSpeechObjektZustandFeld[ObjektNr]=1;
                }
                else
                {
                    // auf gerade pausieren prüfen
                    if (WebSpeechObjektZeigerFeld[ObjektNr].IsPause())
                    {
                        ReturnWert=3;

                        // und WebSpeechObjektZustandFeld korrigieren
                        WebSpeechObjektZustandFeld[ObjektNr]=3;
                    }
                    else
                    {
                        // auf gerade Laden des Textes prüfen
                        if (WebSpeechObjektZeigerFeld[ObjektNr].IsLoading())
                        {
                            ReturnWert=6;

                            // und WebSpeechObjektZustandFeld korrigieren
                            WebSpeechObjektZustandFeld[ObjektNr]=6;
                        }
                    }
                }

                // mehr Möglichkeiten gibts nicht im SDK !
            }
        }
    }

    return ReturnWert;
}

// + + + + + WebSpeech-Objekt erzeugen
//      benötigt WebSpeechInstalliert
function WebSpeech_HTMLObjektErzeugen(    ObjektNr,
                                           AbstandVomLinkenFensterrandInPixel,
                                           AbstandVomOberenFensterrandInPixel,
                                           BreiteInPixel,
                                           HoeheInPixel,
                                           HintergrundAbdecken,

```



```

        HintergrundFarbe,
        Mund_Bewegen,
        Mund_Umrandung_Farbe,
        Mund_Rachen_Farbe,
        Mund_Zunge_Farbe,
        Mund_Zaehne_Farbe,
        Mund_Lippen_Farbe,
        AutoStart,
        Text_Bewegen,
        Text_Farbe,
        Text_Schatten_Farbe,
        Text_Position,
        Text_Hoehe,
        Text                // nur String und keine Url !!
    )

{
    var ReturnWert=false;                // Annahme : Objekt nicht installiert
    var ObjektZustandWert=0;

    if (!WebSpeech_AufErzeugtesHTMLObjektPruefen(ObjektNr))
    {
        // Objekt existiert noch nicht, also erzeugen

        var EvalKommando="";

        var MundFarbKette =  Mund_Umrandung_Farbe + ','
                            + Mund_Rachen_Farbe   + ','
                            + Mund_Zunge_Farbe    + ','
                            + Mund_Zaehne_Farbe   + ','
                            + Mund_Lippen_Farbe;

        var HintergrundFarbKette = HintergrundFarbe;

        var TextFarbKette = Text_Farbe + ',' + Text_Schatten_Farbe;

        if (WebSpeechInstalliert > 0)
        {
            // Netscape aktiv und Plugin ist installiert

            // WebSpeech-Objekt im DIV erzeugen mit Ereignissteuerung aktivieren
            // wobei die EVENT-Angaben identisch sein müssen mit den
            // JavaScript-Ereignis-Routinen
            document.write(
                '<DIV ID="DIV_ID_WebSpeech' + ObjektNr.toString() + '"
                + ' STYLE="position: absolute;'
                + ' left:' + AbstandVomLinkenFensterrandInPixel + 'px;'
                + ' top:' + AbstandVomOberenFensterrandInPixel + 'px;'
                + ' width:' + BreiteInPixel + 'px;'
                + ' height:' + HoeheInPixel + 'px;'
                + '""
                + '>\n'
                + '<EMBED'
                + ' NAME="EMBED_ID_WebSpeech' + ObjektNr.toString() + '"\n'
                + ' TYPE="application/x-WebSpeech"\n'
                + ' WIDTH=' + BreiteInPixel + '"\n'
                + ' HEIGHT=' + HoeheInPixel + '"\n'
                + ' PARAM_AUTOSTART=' + AutoStart + '"\n'
                + ' PARAM_IMMEDIATE="1"\n'
                + ' PARAM_TEXT=' + Text + '"\n'
                + ' PARAM_MOUTHCOLOR=" + MundFarbKette + '"\n'
                + ' PARAM_MOUTHANIMATION=" + Mund_Bewegen + '"\n'
                + ' PARAM_TEXTANIMATION=" + Text_Bewegen + '"\n'
                + ' PARAM_TEXTCOLOR=" + TextFarbKette + '"\n'
                + ' PARAM_TEXTPOSITION=" + Text_Position + '"\n'
                + ' PARAM_TEXTSIZE=" + Text_Hoehe + '"\n'
                + ' PARAM_BACKGROUNDCOLOR="
                + HintergrundFarbKette + '"\n'
                + ' PARAM_OPAQUE=" + HintergrundAbdecken + '"\n'
                + ' EVENT_ONSTARTSPEAKING="DoOnStartSpeaking'
                + ObjektNr.toString() + '(Reason)"\n'
                + ' EVENT_ONSTOPSPEAKING="DoOnStopSpeaking'
                + ObjektNr.toString() + '(Reason)"\n'
                + ' EVENT_ONPAUSESPEAKING="DoOnPauseSpeaking'
                + ObjektNr.toString() + '(Reason)"\n'
                + ' EVENT_ONRESUMESPEAKING="DoOnResumeSpeaking'

```



```

        + ObjektNr.toString() + '(Reason)'"
        + '>\n'
        + '</EMBED>\n'
        + '</DIV>\n'
    );

    // und Zeiger merken
    WebSpeechObjektZeigerFeld[ObjektNr]=
        eval('window.document.EMBED_ID_WebSpeech' + ObjektNr.toString());

    ReturnWert=true;
}
else
{
    if (WebSpeechInstalliert < 0)
    {
        // IE aktiv und ActiveX ist installiert

        // WebSpeech-Objekt im DIV erzeugen
        // anstelle von EVENT muss VisualBasicScript verwendet werden "
        document.write(
            '<DIV ID="DIV_ID_WebSpeech' + ObjektNr.toString() + '"
            + ' STYLE="position: absolute;'
            + 'left:' + AbstandVomLinkenFensterrandInPixel + 'px;'
            + 'top:' + AbstandVomOberenFensterrandInPixel + 'px;'
            + 'width:' + BreiteInPixel + 'px;'
            + 'height:' + HoeheInPixel + 'px;'
            + '"
            + '>\n'
            + '<OBJECT ID="OBJECT_ID_WebSpeech' + ObjektNr.toString() + '"\n'
            + ' CLASSID="" + CLASS_ID + '"\n'
            + ' WIDTH="" + BreiteInPixel + '"\n'
            + ' HEIGHT="" + HoeheInPixel + '"\n'
            + '>\n'
            + '<PARAM NAME="AUTOSTART"
            + " VALUE="" + AutoStart + '">\n'
            + '<PARAM NAME="IMMEDIATE"
            + " VALUE="1">\n'
            + '<PARAM NAME="TEXT"
            + " VALUE="" + Text + '">\n'
            + '<PARAM NAME="MOUTHCOLOR"
            + " VALUE="" + MundFarbKette + '">\n'
            + '<PARAM NAME="MOUTHANIMATION"
            + " VALUE="" + Mund_Bewegen + '">\n'
            + '<PARAM NAME="TEXTANIMATION"
            + " VALUE="" + Text_Bewegen + '">\n'
            + '<PARAM NAME="TEXTCOLOR"
            + " VALUE="" + TextFarbKette + '">\n'
            + '<PARAM NAME="TEXTPOSITION"
            + " VALUE="" + Text_Position + '">\n'
            + '<PARAM NAME="TEXTSIZE"
            + " VALUE="" + Text_Hoehe + '">\n'
            + '<PARAM NAME="BACKGROUNDCOLOR"
            + "VALUE="" + HintergrundFarbKette + '">\n'
            + '<PARAM NAME="OPAQUE"
            + " VALUE="" + HintergrundAbdecken + '">\n'
            + '</OBJECT>\n'
            + '</DIV>\n'
        );

        // und Zeiger merken
        WebSpeechObjektZeigerFeld[ObjektNr]=
            eval('document.OBJECT_ID_WebSpeech' + ObjektNr.toString());

        ReturnWert=true;
    }
}

if (ReturnWert)
{
    // Dimension des Objektes merken
    WebSpeechObjektBreiteFeld[ObjektNr] = BreiteInPixel;
    WebSpeechObjektHoeheFeld[ObjektNr] = HoeheInPixel;

    // Positionen des Objektes merken zum eventuellen Löschen auf Bildschirm

```



```

        WebSpeechObjektLeftFeld[ObjektNr]           = AbstandVomLinkenFensterrandInPixel;
        WebSpeechObjektTopFeld[ObjektNr]           = AbstandVomOberenFensterrandInPixel;

        // und Textquelle "String" und Text merken
        WebSpeechObjektTextQuelleFeld[ObjektNr]      = true;    // true = String, false = Url
        WebSpeechObjektTextOderUrlFeld[ObjektNr]     = Text;

        // Objektzustand ermitteln: aktualisiert WebSpeechObjektZustandFeld[]
        // Funktionswert egal, da nur das Zustandfeld gefüllt werden muss
        ObjektZustandWert=WebSpeech_HTMLObjektZustandErmitteln(ObjektNr);
    }

    return ReturnWert;
}

// + + + + + WebSpeech-Objekt vom Bildschirm verschwinden lassen
function WebSpeech_HTMLObjektEinAusblenden(ObjektNr, AnzeigeArt)
// liefert true, wenn Objekt ausgeblendet wurde
// sonst false, auch wenn Parameter falsch
// ObjektNr >=0
// AnzeigeArt true, so einblenden
// false, so ausblenden
{
    var ReturnWert=false;           // Annahme : Objekt nicht gelöscht
    var EvalKommando="";

    if (WebSpeechObjektZeigerFeld[ObjektNr])
    {
        var Faktor=0;

        if (AnzeigeArt)
        {
            // einblenden
            Faktor=1;
        }
        else
        {
            // ausblenden
            Faktor=-1;
        }

        if (WebSpeechInstalliert > 0)
        {
            // Netscape aktiv und Plugin ist installiert
            // Objekt komplett aus dem Bildschirm schieben

            EvalKommando= 'document.layers.DIV_ID_WebSpeech' + ObjektNr.toString()
                + '.left='
                + (Faktor * WebSpeechObjektLeftFeld[ObjektNr])
                + ';';
            eval(EvalKommando);

            EvalKommando= 'document.layers.DIV_ID_WebSpeech' + ObjektNr.toString()
                + '.top='
                + (Faktor * WebSpeechObjektTopFeld[ObjektNr])
                + ';';
            eval(EvalKommando);

            ReturnWert=true;
        }
        else
        {
            if (WebSpeechInstalliert < 0)
            {
                // IE aktiv und ActiveX ist installiert
                // Objekt komplett aus dem Bildschirm schieben
                EvalKommando= 'document.all.DIV_ID_WebSpeech' + ObjektNr.toString()
                    + '.style.left='
                    + (Faktor * WebSpeechObjektLeftFeld[ObjektNr])
                    + ';';
                eval(EvalKommando);

                EvalKommando= 'document.all.DIV_ID_WebSpeech' + ObjektNr.toString()

```




```

        + '.style.top='
        + ( Faktor * WebSpeechObjektTopFeld[ObjektNr])
        + ';'
    eval(EvalKommando);

    ReturnWert=true;
}

}

return ReturnWert;
}

// + + + + + WebSpeech-Objekt löschen und vom Bildschirm verschwinden lassen
function WebSpeech_HTMLObjektLoeschen(ObjektNr)
// liefert true, wenn Objekt erzeugt wurde
// sonst false, auch wenn Parameter falsch
// Es ist egal, welchen Zustand das Objekt hat
// aber ein gerade gesprochener Text (auch gestapelter) wird zu Ende gesprochen,
// auch wenn das Objekt bereits gelöscht ist
{
    var ReturnWert=false;          // Annahme : Objekt nicht gelöscht

    // Objekt ausblenden auf BS
    ReturnWert = WebSpeech_HTMLObjektEinAusblenden(ObjektNr, false);

    if (ReturnWert)
    {
        // und DANACH die Instanz löschen
        WebSpeechObjektZeigerFeld[ObjektNr] = null;
        WebSpeechObjektBreiteFeld[ObjektNr] = null;
        WebSpeechObjektHoeheFeld[ObjektNr] = null;
        WebSpeechObjektLeftFeld[ObjektNr] = null;
        WebSpeechObjektTopFeld[ObjektNr] = null;
        WebSpeechObjektTextOderUrlFeld[ObjektNr] = null;
        WebSpeechObjektTextQuelleFeld[ObjektNr] = null;    // true = String, false = Url
    }

    return ReturnWert;
}

// + + + + + WebSpeech-Objekt-Text laden als String oder aus Datei, OHNE Sprechen
// benötigt erzeugtes Objekt
function WebSpeech_TextLaden(ObjektNr, Quelle, TextOderUrl)
// Text aus Datei oder Textstring in das Objekt laden
// liefert true, wenn Objekt geladen wurde
// sonst false, auch wenn Parameter falsch oder Objekt nicht installiert oder ein Text bereits geladen wird

// vor dem Laden wird eine bereits aktive Sprachausgabe nicht verändert
// gemutete Sprachausgabe nicht gestartet und nicht zu Ende geführt
// da NUR der User demuten kann

// Es gilt für das Laden eines Textes Direkt nach Erzeugung des Objektes per
// WebSpeech_HTMLObjektErzeugen
// also per Javascript-Folge

// WebSpeech_HTMLObjektErzeugen(...);
// WebSpeech_TextLaden(...);          aus einer DATEI

// Sprachausgabe MIT dem Erzeugen erfolgt NICHT !!!!

// bzw. WebSpeech_TextLaden(...);          aus einer String

// Sprachausgabe MIT dem Erzeugen erfolgt ANHAND des TEXTES laut
// WebSpeech_TextLaden() !!

// Abhilfe kommt nur durch Einfügen eines Tastaturereignisses vor WebSpeech_TexLaden
// z.B. alert();

// ObjektNr : >=0
// Quelle : 0 so String in TextOderUrl
//          1 so Url in TextOderUrl
// TextOderUrl : siehe Quelle
// Für String gilt: Kann leer sein, also nur zum

```




```

        return (Wert >= 0);
    }

// + + + + + WebSpeech-Objekt-Text Sprechen
//                               benötigt erzeugtes Objekt und geladenen Text
function WebSpeech_AktuellGeladenenTextSprechen(ObjektNr, Warten)
// Die Funktion hat nur einen Effekt, wenn sie ein vorhergehendes Tastatur- oder Mausereignis erzeugt wurde
// eine stummgeschaltete Sprachausgabe blockiert das Sprechen des Textes, da NUR der User demuten kann !!
// vor dem Sprechen wird eine pausierte Sprachausgabe weitergeführt, dann Verhalten laut Warten

//      liefert      true, wenn Objekt gesprochen wird
//                  sonst false, wenn Parameter falsch oder Objekt nicht installiert oder stummgeschaltet
//                  oder Text gerade geladen wird
//
//      ObjektNr      : >=0
//      Warten         : 0      Sprachausgabe sofort starten und bereits aktive
//                               abbrechen
//                               1      Sprachausgabe erst starten, wenn aktive beendet ist
{
    var Wert=-1;          // Annahme: Sprechen ohne Erfolg
    var ObjektZustandWert=0;

    if (WebSpeechObjektZeigerFeld[ObjektNr])          // Objekt hat Instanz
    {
        // Objektzustand ermitteln: aktualisiert WebSpeechObjektZustandFeld[], dann auslesen
        ObjektZustandWert=WebSpeech_HTMLObjektZustandErmitteln(ObjektNr);

        if (ObjektZustandWert < 5)
        {
            //      Sprachausgabe zum Objekt ist
            //      nicht stummgeschaltet, also neue Sprachausgabe möglich
            //      Hinweis: Aufheben der Stummschaltung nicht nur durch Javascript
            //      sondern nur durch Useraktion möglich
            //      UND kein Text ist gerade am Laden

            // Objektzustand ermitteln: aktualisiert WebSpeechObjektZustandFeld[], dann auslesen
            ObjektZustandWert=WebSpeech_HTMLObjektZustandErmitteln(ObjektNr);

            if (ObjektZustandWert == 3)
            {
                //      pausierende Sprachausgabe weiterführen
                WebSpeechObjektZeigerFeld[ObjektNr].ResumeSpeaking();
            }

            if ( Warten == 0 )
            {
                // kein Warten, also aktive Sprache abbrechen und neuen Text sprechen
                Wert = WebSpeechObjektZeigerFeld[ObjektNr].StartSpeakingImmediate();
            }
            else
            {
                // erst warten dann Text sprechen
                Wert = WebSpeechObjektZeigerFeld[ObjektNr].StartSpeaking();
            }
        }
    }

    // Objektzustand ermitteln: aktualisiert WebSpeechObjektZustandFeld[]
    //      Funktionswert egal, da nur das Zustandfeld gefüllt werden muss
    ObjektZustandWert=WebSpeech_HTMLObjektZustandErmitteln(ObjektNr);

    return (Wert >= 0);
}

// + + + + + WebSpeech-Objekt-Text temporär laden und sprechen
//                               alten Text retten, neuen Text laden und sprechen, alten Text laden
function WebSpeech_TextTemporaerLadenUndSprechen(ObjektNr, Quelle, TextOderUrl, Warten)
//                               benötigt erzeugtes Objekt
//      ObjektNr      :      >=0
//      Quelle         :      0 so String in TextOderUrl
//                               1 so Url   in TextOderUrl
//      TextOderUrl    :      siehe Quelle
//                               Für String gilt: Kann leer sein, also nur zum LÖSCHEN des
//                               Objekt-Text-Puffers, damit Useraktion

```



```

//                                auf Control-Leiste keine Ausgabe auslöst
//                                Für Url gilt: darf nicht leer sein !!!
//                                Der Dateiname ist beliebig, darf aber nicht mit #
//                                beginnen
//                                (wird nicht geprüft !)
//                                Als Url sind z.B. möglich
//                                www.test.de/test.txt
//                                file:///c:/test.txt
//                                test.txt
//                                im gleichen Verzeichnis wie das
//                                HTML-Dokument
//                                text/test.txt im Unterverzeichnis zum
//                                Verzeichnis
//                                des HTML-Dokumentes
// Warten : 0 Sprachausgabe sofort starten und bereits aktive
//          1 abbrechen
//          1 Sprachausgabe erst starten, wenn aktive beendet ist
{
    var ReturnWert=false;

    // retten der aktuellen Textquelle und des Textes
    var RetteTextQuelle = WebSpeechObjektTextQuelleFeld[ObjektNr];
    var RetteTextOderUrl = WebSpeechObjektTextOderUrlFeld[ObjektNr];

    // Text laden
    ReturnWert= WebSpeech_TextLaden(ObjektNr, Quelle, TextOderUrl);
    if (ReturnWert)
    {
        // und sprechen
        ReturnWert=WebSpeech_AktuellGeladenenTextSprechen(ObjektNr, Warten)
        if (ReturnWert)
        {
            // und geretten Text wieder laden, danach ist leider Text nicht sprechbar
            ReturnWert=WebSpeech_TextLaden(ObjektNr, RetteTextQuelle, RetteTextOderUrl);
        }
    }

    return ReturnWert;
}

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false; // NS 4.x
var ie = document.all ? true : false;

// ++++++ auf Webspeech-Objekt erzeugen prüfen
var WebSpeechInstalliert = WebSpeechInstallationPruefen();
//      0 für Webspeech-Plugin ist nicht installiert
//      im Browser (egal ob NS oder IE)
//      oder IE bzw. NS nicht aktiv
//      > 0 für Webspeech-Plugin ist installiert im NS 4.x
//      und NS 4.x aktiv
//      < 0 für Webspeech-Plugin ist installiert im IE
//      und IE aktiv

var WebSpeech_MenuAnsage_Text = 'das ist ein test';

if (WebSpeechInstalliert !=0)
{
    WebSpeech_TextTemporaerLadenUndSprechen( WebSpeechStandardObjektNr,
    0,
    WebSpeech_MenuAnsage_Text,
    1
    );
}

```

5.3.3. Webspeech-Sprechtags (Auswahl)

5.3.3.1. Webspeech-Sprechtags unter Webspeech 4

Tags tauchen im Text auf, der gesprochen werden soll
werden in \ \ kodiert
Fettmarkiertes sind Pflichtkodierungen

Lautstärke: relativ zur aktuellen Lautstärkeeinstellung laut Windows-Lautstärke-Regelung



			0 bis 65535
			0 stumm
			65535 maximal
Pause	<code>\PAU=value\</code>	value	Integer in Millisekunden ab 0
Reset	<code>\RST\</code>		von Sprechgeschwindigkeit und Lautstärke
Sprechgeschwindigkeit	<code>\SPD=value\</code>	value	Integer Anzahl der Worte pro Minute 80 bis 160
Sprecherwechsel	<code>\MOD=value\</code>	value	Integer 0 Johanna 1 Martin 2 Bill 3 Silke ab 4 nur wenn weitere installierte Stimmen vorhanden sind

5.3.3.2. Webspeech-Sprechtags in Webspeech 4

Sprechtags aus Webspeech-Versionen unter 4 werden dann nicht verarbeitet, wenn
 Webspeech 4 einen anderen Syntax hat (außer \, das als # interpretiert wird)
 Wert nicht in der Schnittmenge der Wertebereiche von Webspeech 4 und der Vorgängerversionen liegt
 Einheit des Wertes nicht in Webspeech 4 bekannt ist bzw. fehlt, aber in Webspeech 4 kodiert werden muss
 Tags tauchen im Text auf, der gesprochen werden soll
 werden in # # kodiert
 Fettmarkiertes sind Pflichtkodierungen

Lautstärke relativ zur aktuellen Lautstärke laut Stimmvorgabe

	<code>#VOL=valuedB#</code>	value	Floating Point in Dezibel -100 bis +12 Kleinste Einheit ist 0.1
	<code>#VOL=value%#</code>	value	Integer in Prozent 10 bis 400 100 entspricht 0dB 200 entspricht +6dB 50 entspricht -6dB
Pause	<code>#PAU=value#</code>	value	Integer in Millisekunden 0 bis 5000

Rücksetzen von Sprechgeschwindigkeit und Lautstärke

`#RST#`

Sprechgeschwindigkeit relativ zur aktuellen Geschwindigkeit laut Stimmvorgabe

	<code>#SPD=value%#</code>	value	Integer in Prozent 10 bis 400
Stimmwechsel	<code>#VOICE=xxx.yy.zzzzzzz#</code>	xxx	Herstellerkürzel 3 Zeichen z.B. LGX
		yy	Sprache 2 Zeichen z.B. DE
		zzzzzzz	Stimmname max. 8 Zeichen z.B. BILL

Punkte müssen kodiert werden

Standardstimmen sind



			LGX.DE.BILL
			LGX.DE.SABINE
			LGX.DE.MANDY
			LGX.DE.KATHI
			LGX.DE.SILKE
			LGX.DE.MICHAEL
			LGX.DE.MARTIN
Text	Text buchstabieren (Zahlen ziffernweise)		
		#CTX=SPELL#	Einschalten des Buchstabierens
		#CTX#	Abschalten des Buchstabierens
Wort	Betonung für Wort abschalten		
		#NOACC#	vor dem Wort kodieren
Wort	fremdes Wort der Sprachausgabe als Wortart bekanntgeben Variante 1		
		#POS=kette#	vor dem Wort kodieren
		kette	NOUN Substantiv
			NAME Eigenname
			VERB Verb
			ADJ Adjektiv
			NUMB Zahlwort
			PRON Pronomen z.B. sie
			ART Artikel
			PREP Präposition z.B. nach
			ADV Adverb z.B. damals
			CONJ Konjunktion z.B. weil, und
			PART Partikel z.B. vielleicht, schon
			INTJ Interjektion z.B. Ach! als Ausruf
Wort	fremdes Wort der Sprachausgabe als Wortart bekanntgeben Variante 2		
		#FW#	vor dem Wort kodieren
			Dauer der Laute verkürzen wenn Wort ist
			Präposition z.B. nach
			Artikel
			Konjunktion z.B. weil, und
Wort	Vokal eines nachfolgenden Wortes länger sprechen und Wort mit Betonung sprechen		
		#EMPH#	vor dem Wort kodieren
Wortfolge	Betonung abschalten bis Satzende		
		#LASTACC#	ab Wort dahinter bis Satzende

5.4. Beispiel für windowseigenes Active-X-Control – Analoge Uhr

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-



Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen. Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899
Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit Kompression genutzt wird z.B. bei
onlick-Handler auf IMG
klick ins Fenster per aktivem Popup

Der Absturz ist "read" -Fehler von immer ein und derselben Speicherstelle.
User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.
Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:
Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das
Patchproblem auftritt (User muss sich Telefonnummer besorgen)
Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,
z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer
von Microsoft besorgen muss bzw. zu besorgen hat, wird der User
IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.
(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,

bricht der Browser das Dokument mit .show() ab (Scriptfehler).

Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende Meldung angezeigt (in der Informationsleiste):

'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'

Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:

Der Popupblocker hat ein Populfenster geblockt. Sie können den Popupblocker deaktivieren oder Pops temporär zulassen, indem Sie auf die Informationsleiste klicken.

Die Realität zur obigen Meldung ist völlig anders:

Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter

Popupblocker einschalten

weitere Informationen

jedoch keine Möglichkeit wie laut Bedeutung

Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.

Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)

Der Popupblocker des IE bemeckert aber NUR Webseite, die das Popup erzeugt.

Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.

Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).

Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.

Der Popupblockerfehler verändert die Eventverwaltung:

Es werden u.a. ignoriert

onfocus

onblur

onfocusin

onfocusout

und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.




```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')           // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
{document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupefehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
zwischen Register in einem IE-Fenster
zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
.setActive() ist Teilmenge von .focus(): nur das aktiv setzen
funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
z.B. garnicht mehr sichtbar, oder Timer nicht verwendet.
Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtsstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist. Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.
Es muss also ERST per Mausclick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:

Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X-Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.



Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tablular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformatvorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
(http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind

Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)

http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere

Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)



Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Umeine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:•

[http://www.microsoft.com/technet/security/bulletin/ms99-](http://www.microsoft.com/technet/security/bulletin/ms99-037.msp)

[037.msp](http://www.microsoft.com/technet/security/bulletin/ms99-037.msp)(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)

<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>

(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)

240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen{0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen:• 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.



IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Popupmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Popupmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • <http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm> (<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

```
<HTML>
<HEAD>
```

```
<SCRIPT LANGUAGE="JScript">
<!--
```

```
// Analoguhr
// nur IE
// Es werden Active-X-Controls verwendet (DirectX)
```

```
// ##### veränderbare Variablen #####
```

```
// +++++ Uhr +++++
```

```
// Dimension der Uhr
var AnalogUhr_BreiteUndHoehe=330; // >=140 und <=330
```

```
// Lage der Uhr, alles Stringangaben
var AnalogUhr_Top=20;
var AnalogUhr_Left=20;
```

```
// +++++ Ziffern +++++
```

```
var AnalogUhr_Ziffern_Erzeugen=true; // false für keine Ziffern
var AnalogUhr_Ziffer_Fett=false; // false für nicht fett
```

```
var AnalogUhr_Ziffer_Standard='.';
var AnalogUhr_Ziffer_Stunde_Viertel='*';
var AnalogUhr_Ziffer_Stunde_Halbe='+';
var AnalogUhr_Ziffer_Stunde_DreiViertel='*';
```



```

var AnalogUhr_Ziffer_Stunde_Voll='#';

var AnalogUhr_Ziffern_Satz=2; // 0 für frei definiert per o.g. Variablen
    // 1 für römisch (vordefiniert)
    // 2 für normal (vordefiniert)

// Farbe aller Ziffern, alles Stringangaben
var AnalogUhr_Ziffer_Farbe_RotAnteil='255';
var AnalogUhr_Ziffer_Farbe_GruenAnteil='255';
var AnalogUhr_Ziffer_Farbe_BlauAnteil='255';

// +++++ Zeiger +++++
// Sekundenzeiger erzeugen
var AnalogUhr_Zeiger_Sekunden_Erzeugen=true; // false für keinen Sekundenzeiger

// Zeigerform
var AnalogUhr_Zeiger_Flaeche=true; // false für hohl (falls überhaupt renderbar)

// Farbe des Sekunden-Zeigers, alles Stringangaben
var AnalogUhr_Zeiger_Sekunden_Farbe_RotAnteil='255';
var AnalogUhr_Zeiger_Sekunden_Farbe_GruenAnteil='0';
var AnalogUhr_Zeiger_Sekunden_Farbe_BlauAnteil='0';

// Farbe des Minuten-Zeigers, alles Stringangaben
var AnalogUhr_Zeiger_Minuten_Farbe_RotAnteil='170';
var AnalogUhr_Zeiger_Minuten_Farbe_GruenAnteil='255';
var AnalogUhr_Zeiger_Minuten_Farbe_BlauAnteil='255';

// Farbe des Stunden-Zeigers, alles Stringangaben
var AnalogUhr_Zeiger_Stunden_Farbe_RotAnteil='204';
var AnalogUhr_Zeiger_Stunden_Farbe_GruenAnteil='238';
var AnalogUhr_Zeiger_Stunden_Farbe_BlauAnteil='255';

// +++++ Zeiger-Ursprung +++++
// Farbe des Zeiger-Ursprunges, alles Stringangaben
var AnalogUhr_Zeiger_Ursprung_Farbe_RotAnteil='255';
var AnalogUhr_Zeiger_Ursprung_Farbe_GruenAnteil='255';
var AnalogUhr_Zeiger_Ursprung_Farbe_BlauAnteil='0';

// +++++ Hintergrund der Uhr +++++
// Farbe des Hintergrundes, alles Stringangaben
var AnalogUhr_Hintergrund_Farbe_RotAnteil='26';
var AnalogUhr_Hintergrund_Farbe_GruenAnteil='78';
var AnalogUhr_Hintergrund_Farbe_BlauAnteil='133';

// Farbe des Hintergrund-Rahmens, alles Stringangaben
var AnalogUhr_HintergrundRahmen_Farbe_RotAnteil='200';
var AnalogUhr_HintergrundRahmen_Farbe_GruenAnteil='200';
var AnalogUhr_HintergrundRahmen_Farbe_BlauAnteil='200';
// -->
</SCRIPT>

<SCRIPT LANGUAGE="VBScript">

' ##### Steuerung der Uhr anhand ActiveX-Objekte #####

'++++ Variablen
' Variablen aus JScript sind in VBScript referenzierbar !

Dim AnalogUhr_Zeit
Dim AnalogUhr_Zeit_Minute
Dim AnalogUhr_Zeit_Sekunde
Dim AnalogUhr_Zeit_Stunde

'++++ aktuelle Zeit holen
Sub ZeitHolen
'aktuelle Zeit holen
AnalogUhr_Zeit = time
AnalogUhr_Zeit_Sekunde = second(AnalogUhr_Zeit)
AnalogUhr_Zeit_Minute = minute(AnalogUhr_Zeit)
AnalogUhr_Zeit_Stunde = hour(AnalogUhr_Zeit)

```



```

'prüfen ob aktuelle AnalogUhr_Zeit_Stunde > 12, wenn ja dann auf 12-Stundenbasis dezimieren
if AnalogUhr_Zeit_Stunde > 12 then
    AnalogUhr_Zeit_Stunde = AnalogUhr_Zeit_Stunde - 12
end if
end Sub

'++++ Zeiger bewegen
Sub ZeigerBewegen
'aktuelle Zeit holen
ZeitHolen

'Sekundenzeiger bewegen, falls Sekundenzeiger erzeugt wurde
if AnalogUhr_Zeiger_Sekunden_Erzeugen then
    Call ID_AnalogUhr_Zeiger_Sekunden.Rotate(0,0,6)
end if

'danach den Minutenzeiger
if AnalogUhr_Zeit_Sekunde = 0 then
    Call ID_AnalogUhr_Zeiger_Minuten.Rotate(0,0,6)

'danach den Stundenzeiger
if AnalogUhr_Zeit_Minute mod 2 = 0 then
    call ID_AnalogUhr_Zeiger_Stunden.Rotate(0,0,1)
end if
end if
End Sub

'++++ Uhrwerk-Objekt mit der VB-Routine verbinden, also Feder ins Uhrwerk einbauen
'    und Uhrwerk laufen lassen, also Zeiger bewegen
' ID_AnalogUhr_UhrWerk ist Objekt
' OnInit ist Ereignis
' _ ist Kodierungsvorschrift
Sub ID_AnalogUhr_UhrWerk_OnInit
'Timer einrichten für das Objekt
'    ruft ZeigerBewegen auf
call ID_AnalogUhr_UhrWerk("ID_AnalogUhr_Timer").at(1.000, "ZeigerBewegen", -1,1.000, 1)
End Sub

'++++ Mit Laden des Dokumentes aktivierte Routine: Uhrwerk initialisieren
' window ist Objekt
' onload ist Ereignis
' _ ist Kodierungsvorschrift
Sub window_onload
'aktuelle Zeit holen
ZeitHolen

'Objekte in Erstposition laut aktuelle Zeit setzen: Umrechnung bezogen auf 360 Grad
' Positionierung per Methode des jeweiligen Objektes
'    AnalogUhr_Zeit_Sekunde*6 ist maximal 60 * 6 also 360 Grad
if AnalogUhr_Zeiger_Sekunden_Erzeugen then
    call ID_AnalogUhr_Zeiger_Sekunden.Rotate(0,0,(AnalogUhr_Zeit_Sekunde*6) - 90)
end if

call ID_AnalogUhr_Zeiger_Minuten.Rotate(0,0,(AnalogUhr_Zeit_Minute*6) - 90)

'    AnalogUhr_Zeit_Stunde*360/12 ist maximal 12*360/12 also 360 Grad
call ID_AnalogUhr_Zeiger_Stunden.Rotate(0,0,(AnalogUhr_Zeit_Stunde*360/12) - 90)

'    AnalogUhr_Zeit_Minute/2 ist maximal 60/2 also 30 Grad
call ID_AnalogUhr_Zeiger_Stunden.Rotate(0,0,int(AnalogUhr_Zeit_Minute/2))

'und Uhrwerk ticken lassen per Timer starten (.Play)
'    und damit Zeiger bewegen laut mit Timer verbundenen funktion ZeigerBewegen
'    siehe ID_AnalogUhr_UhrWerk_OnInit
call ID_AnalogUhr_UhrWerk("ID_AnalogUhr_Timer").Play
end Sub
</SCRIPT>

<SCRIPT LANGUAGE="JScript">
<!--

// ##### interne Variablen #####

```



```

// ID der ActiveX-Controls
var Objekt_ClassID_Uhrwerk="";
var Objekt_ClassID_UhrLayout="";

// Maximalwert für Veränderung von Breite und Höhe
var Uhr_MaximaleVeränderung_BreiteUndHoehe=0;

// Zeigerabweichung für alle Zeiger
var ZeigerAbweichung_Horizontal=0;

// Ovalabweichung für Hintergrund
var Hintergrund_OvalVerschiebung_HorizontalUndVertikal=0;

// ##### Funktionen zur Erzeugung der Uhr #####

// ++++++ Uhrwerk-Objekt ++++++

function UhrWerk_Objekt_Erzeugen()
{
    // Das Uhrwerk ist der Motor der Uhr und wird NICHT gerendert
    document.write( '<OBJECT ID="ID_AnalogUhr_UhrWerk"'
        + ' CLASSID="'+ Objekt_ClassID_Uhrwerk + '"'
        + '>'
        + '</OBJECT>'
    );
}

// ++++++ Uhr-Layout-Objekt ++++++

// ----- Uhr-Layout-Objekt: Kopf erzeugen -----

function UhrLayout_ObjektKopf_Erzeugen(IDKette,BreiteAenderung,HoeheAenderung)
// Änderung der Breite bzw. Höhe dient zum optisch-symmetrischen Ausrichten der Uhrteile
// z.B. Zifferänderung von Römisch (wie XII) auf 12 bewirkt andere Breite, also
// muss die Lage des Hintergrundes korrigiert werden, damit optische Symmetrie anliegt.
{
    var UhrLayout_ObjektKopf_Erzeugen_Wert1=0;
    var UhrLayout_ObjektKopf_Erzeugen_Wert2=0;

    // Maximalwerte der Veränderung einhalten
    if (BreiteAenderung > Uhr_MaximaleVeränderung_BreiteUndHoehe)
    {BreiteAenderung = Uhr_MaximaleVeränderung_BreiteUndHoehe;}

    if (HoeheAenderung > Uhr_MaximaleVeränderung_BreiteUndHoehe)
    {HoeheAenderung = Uhr_MaximaleVeränderung_BreiteUndHoehe;}

    // Veränderung vollziehen
    UhrLayout_ObjektKopf_Erzeugen_Wert1=AnalogUhr_BreiteUndHoehe + BreiteAenderung;
    UhrLayout_ObjektKopf_Erzeugen_Wert2=AnalogUhr_BreiteUndHoehe + HoeheAenderung;

    document.write( '<OBJECT ID="'+ IDKette + '"'
        + ' CLASSID="'+ Objekt_ClassID_UhrLayout + '"'
        + ' STYLE="position:absolute;'
        + ' width:' + UhrLayout_ObjektKopf_Erzeugen_Wert1 + ';'
        + ' height:' + UhrLayout_ObjektKopf_Erzeugen_Wert2 + ';'
        + ' top:' + AnalogUhr_Top + ';'
        + ' left:' + AnalogUhr_Left
        + ' "'
        + '>'
    );
}

// ----- Uhr-Layout-Objekt: Hintergrund -----

function UhrLayout_Hintergrund_Erzeugen()
{
    // Oval-Dimension ermitteln
    var Oval_BreiteUndHoehe=AnalogUhr_BreiteUndHoehe-10;

    // Oval-Abweichung horizontal ermitteln
    // Annahme: keine römischen Ziffern
    var OvalAbweichung_Horizontal=ZeigerAbweichung_Horizontal;

    // auf römische Ziffern prüfen

```




```
if (AnalogUhr_Ziffern_Satz == 1)
{OvalAbweichung_Horizontal = ZeigerAbweichung_Horizontal * 3};
```

```
UhrLayout_ObjektKopf_Erzeugen('ID_AnalogUhr_Hintergrund',
    OvalAbweichung_Horizontal,
    ZeigerAbweichung_Horizontal
);
```

```
document.write(
    '<PARAM NAME="Line0001" VALUE="SetLineColor('
    + AnalogUhr_HintergrundRahmen_Farbe_RotAnteil + ','
    + AnalogUhr_HintergrundRahmen_Farbe_GruenAnteil + ','
    + AnalogUhr_HintergrundRahmen_Farbe_BlauAnteil
    + ')
    + '
    + '>'
    + '<PARAM NAME="Line0001" VALUE="SetLineStyle(5)">'
    + '<PARAM NAME="Line0002" VALUE="SetFillStyle(1)">'
    + '<PARAM NAME="Line0003" VALUE="SetFillColor('
    + AnalogUhr_Hintergrund_Farbe_RotAnteil + ','
    + AnalogUhr_Hintergrund_Farbe_GruenAnteil + ','
    + AnalogUhr_Hintergrund_Farbe_BlauAnteil
    + ')
    + '
    + '>'
    + '<PARAM NAME="Line0004" VALUE="SetGradientFill(0,0,'
    + Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
    + Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
    + '
    + '0)'
    + '
    + '>'
    + '<PARAM NAME="Line0005" VALUE="Oval('
    + Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
    + // Verschiebung Ovalkante horizontal bezüglich Zeigerursprung
    + // < 0 nach links, > 0 nach rechts
    + Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
    + // Verschiebung vertikal bezüglich Zeigerursprung
    + // < 0 nach oben, > 0 nach unten
    + Oval_BreiteUndHoehe + ',' // Breite des Ovals
    + Oval_BreiteUndHoehe + ',' // Höhe des Ovals
    + '
    + '0)'
    + '
    + '>'
    + '</OBJECT>'
);
}
```

// ----- Uhr-Layout-Objekt: Ziffern -----

```
function UhrLayout_Ziffern_Erzeugen(ParameterNummerAlsString,Ziffer,Breite,Hoehe)
{
    document.write( ' <PARAM NAME="Line00' + ParameterNummerAlsString + '"
    + ' VALUE="Text(' + Ziffer + ',' + Breite + ',' + Hoehe + ',' + '0' + ')"'
    + '>'
    );
}
```

```
function UhrLayout_Ziffern_Erzeugen()
{
    // +++++ Schriftart festlegen
    var Ziffer_SchriftArt="times new roman"; // nicht ändern !!!

    // +++++ Schrifthöhe berechnen
    var Ziffer_SchriftHoehe=Math.floor(AnalogUhr_BreiteUndHoehe / 10); // nicht ändern !!

    // Fett-Darstellung der Ziffer
    // Annahme: nicht fett
    var ZifferDicke=65; // nicht ändern
    if (AnalogUhr_Ziffer_Fett)
    {ZifferDicke*=10;} // 650 für fett, nicht ändern

    // +++++ Ziffernart ermitteln
```



```
// ---- Annahme: frei definierte Ziffern verwenden
var Ziffer_Standard      =AnalogUhr_Ziffer_Standard;
var Ziffer_Stunde_Viertel =AnalogUhr_Ziffer_Stunde_Viertel;
var Ziffer_Stunde_Halbe   =AnalogUhr_Ziffer_Stunde_Halbe;
var Ziffer_Stunde_DreiViertel=AnalogUhr_Ziffer_Stunde_DreiViertel;
var Ziffer_Stunde_Voll     =AnalogUhr_Ziffer_Stunde_Voll;

// ---- auf Art prüfen
if (AnalogUhr_Ziffern_Satz > 0)
{
    // prüfen auf römisch (Satz 1)
    if (AnalogUhr_Ziffern_Satz==1)
    {
        // nicht ändern
        Ziffer_Standard      =',';
        Ziffer_Stunde_Viertel = 'III';
        Ziffer_Stunde_Halbe   = 'VI';
        Ziffer_Stunde_DreiViertel= 'IX';
        Ziffer_Stunde_Voll     = 'XII';
    }
    else
    {
        // normal (Satz 2), nicht ändern
        Ziffer_Standard      =',';
        Ziffer_Stunde_Viertel = '3';
        Ziffer_Stunde_Halbe   = '6';
        Ziffer_Stunde_DreiViertel= '9';
        Ziffer_Stunde_Voll     = '12';
    }
}

// ---- Ziffern in das Format für Objekt-Erzeugung umwandeln
// Text-Funktion akzeptiert nur Strings innerhalb "" und nicht innerhalb "
Ziffer_Standard      = "" + Ziffer_Standard      + "";
Ziffer_Stunde_Viertel = "" + Ziffer_Stunde_Viertel + "";
Ziffer_Stunde_Halbe   = "" + Ziffer_Stunde_Halbe   + "";
Ziffer_Stunde_DreiViertel= "" + Ziffer_Stunde_DreiViertel + "";
Ziffer_Stunde_Voll     = "" + Ziffer_Stunde_Voll     + "";

// +++++ Layout der Zifferndarstellung ermitteln
//      ist abhängig von der Breite und Höhe der Analoguhr
//      fixiert für Schriftart laut Ziffer_SchriftArt
//      darf im Programmcode nicht geändert werden

// ---- Umrechnungsfaktor ermitteln
var Ziffer_Umrechnungsfaktor=AnalogUhr_BreiteUndHoehe / 210; // nicht ändern !!

// ---- Ziffern für Viertel- und Dreiviertelstunde

// - - - Layoutverschiebung vertikal bei Umrechnungsfaktor 1/1
var Hoehe_ViertelUndDreiviertelStunde = Math.floor(Ziffer_SchriftHoehe / 4); // nicht ändern !!

//      Layoutverschiebung vertikal bei realem Umrechnungsfaktor
Hoehe_ViertelUndDreiviertelStunde=Math.floor(Hoehe_ViertelUndDreiviertelStunde * Ziffer_Umrechnungsfaktor);

// - - - Layoutverschiebung horizontal bei Umrechnungsfaktor 1/1
//      Viertel : Abstand von Ziffer zum Zeigerursprung enthält NICHT Breite der Ziffer
//      Dreiviertel: Abstand von Ziffer zum Zeigerursprung enthält Breite der Ziffer
//      Problem: Breite der Ziffer nicht ermittelbar !
var Breite_ViertelStunde = 75; // nicht ändern
var Breite_DreiViertelStunde = -90; // nicht ändern

//      Layoutverschiebung horizontal bei realem Umrechnungsfaktor
Breite_ViertelStunde =Math.floor(Breite_ViertelStunde * Ziffer_Umrechnungsfaktor);
Breite_DreiViertelStunde=Math.floor(Breite_DreiViertelStunde * Ziffer_Umrechnungsfaktor);

var Breite_Durchschnitt_ViertelUndDreiViertelStunde= Math.abs(Breite_ViertelStunde)
    + Math.abs(Breite_DreiViertelStunde);
Breite_Durchschnitt_ViertelUndDreiViertelStunde=Math.floor(Breite_Durchschnitt_ViertelUndDreiViertelStunde / 2);
var Breite_Drittel_ViertelUndDreiViertelStunde=Math.floor(Breite_Durchschnitt_ViertelUndDreiViertelStunde / 3);
var Breite_Fuenftel_ViertelUndDreiViertelStunde=Math.floor(Breite_Durchschnitt_ViertelUndDreiViertelStunde / 5);

// ---- Ziffern für volle und halbe Stunde
```



```
// - - - Layoutverschiebung vertikal bei Umrechnungsfaktor 1/1
var Hoehe_HalbeStunde=Math.floor(Ziffer_SchriftHoehe / 3); // nicht ändern
var Hoehe_VolleStunde=Hoehe_HalbeStunde;
Hoehe_HalbeStunde+=82; // nicht ändern
Hoehe_VolleStunde+=76; // nicht ändern

// Layoutverschiebung vertikal bei realem Umrechnungsfaktor
Hoehe_HalbeStunde=Math.floor(Hoehe_HalbeStunde * Ziffer_Umrechnungsfaktor);
Hoehe_VolleStunde=Math.floor(Hoehe_VolleStunde * Ziffer_Umrechnungsfaktor);
Hoehe_VolleStunde*=-1;

var Hoehe_Durchschnitt_HalbeUndVolleStunde= Math.abs(Hoehe_HalbeStunde)
+ Math.abs(Hoehe_VolleStunde);
Hoehe_Durchschnitt_HalbeUndVolleStunde=Math.floor(Hoehe_Durchschnitt_HalbeUndVolleStunde / 2);
var Hoehe_Drittel_HalbeUndVolleStunde=Math.floor(Hoehe_Durchschnitt_HalbeUndVolleStunde / 3);
var Hoehe_Fuenftel_HalbeUndVolleStunde=Math.floor(Hoehe_Durchschnitt_HalbeUndVolleStunde / 5);
var Hoehe_Siebtel_HalbeUndVolleStunde=Math.floor(Hoehe_Durchschnitt_HalbeUndVolleStunde / 7);

// - - - Layoutverschiebung horizontal bei Umrechnungsfaktor 1/1
// Problem: Breite der Ziffer nicht ermittelbar und damit Zentrierung nicht ermittelbar !
var Breite_HalbeStunde=0;
var Breite_VolleStunde=0;

// auf römische Ziffern prüfen (Satz 1)
if (AnalogUhr_Ziffern_Satz==1)
{
    Breite_HalbeStunde=-6; // nicht ändern
    Breite_VolleStunde=-11; // nicht ändern
}
else
{
    Breite_HalbeStunde=-3; // nicht ändern
    Breite_VolleStunde=-8; // nicht ändern
}

// Layoutverschiebung horizontal bei realem Umrechnungsfaktor
Breite_HalbeStunde=Math.floor(Breite_HalbeStunde * Ziffer_Umrechnungsfaktor);
Breite_VolleStunde=Math.floor(Breite_VolleStunde * Ziffer_Umrechnungsfaktor);

// ----- Ziffern zwischen viertel, halbe, dreiviertel und volle Stunde

//
// links < 0 | rechts > 0
//          12
// oben < 0 h | a oben < 0
//          g | b
// --- 9 -----+----- 3 ---
//          f | c
// unten > 0 e | d unten > 0
//          6
// links < 0 | rechts > 0
//          |
//
// Spiegelsymmetrie vertikal (Höhe)
// h a
// g b
// f c
// e d
//
// Spiegelsymmetrie horizontal (Breite)
// a d
// e h
// b c
// f g
//
// - - - Layoutverschiebung horizontal
var a_d_Breite = Breite_Drittel_ViertelUndDreiViertelStunde
+ Breite_Fuenftel_ViertelUndDreiViertelStunde; // rechts
var h_e_Breite = -1 * a_d_Breite; // links
var b_c_Breite = Breite_Drittel_ViertelUndDreiViertelStunde * 2;
b_c_Breite += Breite_Fuenftel_ViertelUndDreiViertelStunde; // rechts
var g_f_Breite = -1 * b_c_Breite; // links
```



```

// - - - Layoutverschiebung vertikal
var a_h_Hoehe = Hoehe_Drittel_HalbeUndVolleStunde * 2;
a_h_Hoehe += Hoehe_Fuenftel_HalbeUndVolleStunde;
a_h_Hoehe *= -1; // oben
var d_e_Hoehe = -1 * a_h_Hoehe; // unten
var b_g_Hoehe = Hoehe_Drittel_HalbeUndVolleStunde
+ Hoehe_Siebtel_HalbeUndVolleStunde;
b_g_Hoehe *= -1; // oben
var c_f_Hoehe = -1 * b_g_Hoehe; // unten

// +++++ Layout der Zifferndarstellung erzeugen

// ----- Objektkopf erzeugen
UhrLayout_ObjektKopf_Erzeugen('ID_AnalogUhr_Ziffern',0,0);

// ----- Font als Parameter des Objektes erzeugen
document.write(
    '<PARAM NAME="Line0001" VALUE="SetLineColor(0, 0, 0)">'
    + '<PARAM NAME="Line0002" VALUE="SetLineStyle(0)">'
    + '<PARAM NAME="Line0003" VALUE="SetFillStyle(1)">'
    + '<PARAM NAME="Line0004" VALUE="SetFillColor('
    + AnalogUhr_Ziffer_Farbe_RotAnteil + ','
    + AnalogUhr_Ziffer_Farbe_GruenAnteil + ','
    + AnalogUhr_Ziffer_Farbe_BlauAnteil
    + ') '
    + '"" '
    + '>'
    + '<PARAM NAME="Line0005" VALUE="SetFont('
    + Ziffer_SchriftArt + ','
    + Ziffer_SchriftHoehe + ','
    + ZifferDicke + ','
    + '0,0,0'
    + ') '
    + '"" '
    + '>'
);

// ----- Ziffern als Parameter des Objektes erzeugen
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('06',Ziffer_Standard,a_d_Breite,a_h_Hoehe); // a
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('07',Ziffer_Standard,b_c_Breite,b_g_Hoehe); // b
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('08',Ziffer_Stunde_Viertel,
    Breite_ViertelStunde,Hoehe_ViertelUndDreiviertelStunde);
    // Viertel Stunde
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('09',Ziffer_Standard,b_c_Breite,c_f_Hoehe); // c
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('10',Ziffer_Standard,a_d_Breite,d_e_Hoehe); // d
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('11',Ziffer_Stunde_Halbe,
    Breite_HalbeStunde,Hoehe_HalbeStunde); // Halbe Stunde
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('12',Ziffer_Standard,h_e_Breite,d_e_Hoehe); // e
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('13',Ziffer_Standard,g_f_Breite,c_f_Hoehe); // f
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('14',Ziffer_Stunde_DreiViertel,
    Breite_DreiViertelStunde,Hoehe_ViertelUndDreiviertelStunde);
    // DreiViertel Stunde
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('15',Ziffer_Standard,g_f_Breite,b_g_Hoehe); // g
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('16',Ziffer_Standard,h_e_Breite,a_h_Hoehe); // h
UhrLayout_Ziffern_Erzeugen_ParameterErzeugen('17',Ziffer_Stunde_Voll,
    Breite_VolleStunde,Hoehe_VolleStunde); // Volle Stunde

document.write( '</OBJECT>');
}

// ----- Uhr-Layout-Objekt: Zeiger -----

function UhrLayout_Zeiger_Erzeugen(IDKette,
    RotAnteil,GruenAnteil,BlauAnteil,
    Wert1,Wert2,Wert3,Wert4
    )
{
    // +++++ Zeigerform
    // Annahme: Zeiger hohl
    var ZeigerForm=2;
    if (AnalogUhr_Zeiger_Flaeche)
    {ZeigerForm--;} // 1 für Fläche

```



```
// +++++ Layout erzeugen

// ----- Kopf des Objektes
UhrLayout_ObjektKopf_Erzeugen(IDKette, ZeigerAbweichung_Horizontal, 0);

// ----- Zeigerfarben
document.write( '<PARAM NAME="Line0001"'
    + ' VALUE="SetLineColor(' + RotAnteil + ',' + GruenAnteil + ',' + BlauAnteil + ')"'
    + '>'
    );

document.write( '<PARAM NAME="Line0002" VALUE="SetLineStyle(1)">' );

document.write( '<PARAM NAME="Line0003"'
    + ' VALUE="SetFillColor(' + RotAnteil + ',' + GruenAnteil + ',' + BlauAnteil + ')"'
    + '>'
    );

// ----- Zeigerform
document.write( '<PARAM NAME="Line0004" VALUE="SetFillStyle(' + ZeigerForm + ')">' );

// ----- Zeigerposition im Layout
document.write( '<PARAM NAME="Line0005"'
    + ' VALUE="Rect(' + Wert1 + ',' + Wert2 + ',' + Wert3 + ',' + Wert4 + ')"'
    + '>'
    + '</OBJECT>'
    );

// Wert1 Abweichung vom Zeigerursprung vertikal, 0 keine Abweichung, > 0 nach oben, < 0 nach unten
// Wert2 Abweichung vom Zeigerursprung horizontal, 0 keine Abweichung, > 0 nach links, < 0 nach rechts
// Wert3 Zeigerlänge
// Wert4 Zeigerdicke
}
```

```
function UhrLayout_AlleZeiger_Erzeugen()
// Hinweis: Alle Zeiger sollten sich im Ursprung treffen: Beachte Dimension der Uhr !
// Wenn sich alle Zeiger treffen, dann nach Augenmass alle Zeiger
// in der Uhrfläche zentrieren.
// Wenn Zeiger zentriert wurden, dann Zeigerursprung als Ring darstellen und
// den Ursprung zentrieren.
{
// Variablen der Verschiebung
// vertikal: negativer Wert für Verschiebung nach unten (n unten)
// positiver Wert für Verschiebung nach oben (p oben)

// horizontal: negativer Wert für Verschiebung nach links (n links)
// positiver Wert für Verschiebung nach rechts (p rechts)
var Vertikale_Verschiebung=0;
var Horizontale_Verschiebung=0;

// +++++ Sekundenzeiger

// ----- Länge
// dient als Basis für die Ermittlung der Längen vom Minuten- und Stundenzeiger
var ZeigerLaenge=Math.floor(AnalogUhr_BreiteUndHoehe/2)-15;
var ZeigerLaenge_Differenz=Math.floor(ZeigerLaenge / 6);

// ----- prüfen ob Sekundenzeiger erzeugt werden soll
// Sekundenzeiger
if (AnalogUhr_Zeiger_Sekunden_Erzeugen)
{
// Sekundenzeiger soll erzeugt werden

// ----- Verschiebung
// auf römische Ziffern (Satz 1) prüfen
if (AnalogUhr_Ziffern_Satz==1)
{
Vertikale_Verschiebung=1; // nicht verändern
Horizontale_Verschiebung=0; // nicht verändern
}
else
{
Vertikale_Verschiebung=-2; // nicht verändern
}
```



```

    Horizontale_Verschiebung=0; // nicht verändern
}

// ----- Zeiger erzeugen
UhrLayout_Zeiger_Erzeugen('ID_AnalogUhr_Zeiger_Sekunden',
    AnalogUhr_Zeiger_Sekunden_Farbe_RotAnteil,
    AnalogUhr_Zeiger_Sekunden_Farbe_GruenAnteil,
    AnalogUhr_Zeiger_Sekunden_Farbe_BlauAnteil,
    Vertikale_Verschiebung,Horizontale_Verschiebung,
    ZeigerLaenge,0 // Zeigerdicke ist 0 für Standarddicke
);

}

// +++++ Minutenzeiger

// ----- Zeigerlänge
ZeigerLaenge=ZeigerLaenge_Differenz;

// ----- Verschiebung
//    auf römische Ziffern (Satz 1) prüfen
if (AnalogUhr_Ziffern_Satz==1)
{
    Vertikale_Verschiebung=1; // nicht verändern
    Horizontale_Verschiebung=1; // nicht verändern
}
else
{
    Vertikale_Verschiebung=3; // nicht verändern
    Horizontale_Verschiebung=0; // nicht verändern
}

// ----- Zeiger erzeugen
UhrLayout_Zeiger_Erzeugen('ID_AnalogUhr_Zeiger_Minuten',
    AnalogUhr_Zeiger_Minuten_Farbe_RotAnteil,
    AnalogUhr_Zeiger_Minuten_Farbe_GruenAnteil,
    AnalogUhr_Zeiger_Minuten_Farbe_BlauAnteil,
    Vertikale_Verschiebung,Horizontale_Verschiebung,
    ZeigerLaenge,1 // Zeigerdicke ist 1
);

// +++++ Stundenzeiger

// ----- Zeigerlänge
ZeigerLaenge=ZeigerLaenge_Differenz;

// ----- Verschiebung
//    auf römische Ziffern (Satz 1) prüfen
if (AnalogUhr_Ziffern_Satz==1)
{
    Vertikale_Verschiebung=0; // nicht verändern
    Horizontale_Verschiebung=-1; // nicht verändern
}
else
{
    Vertikale_Verschiebung=1; // nicht verändern
    Horizontale_Verschiebung=-1; // nicht verändern
}

// ----- Zeiger erzeugen
UhrLayout_Zeiger_Erzeugen('ID_AnalogUhr_Zeiger_Stunden',
    AnalogUhr_Zeiger_Stunden_Farbe_RotAnteil,
    AnalogUhr_Zeiger_Stunden_Farbe_GruenAnteil,
    AnalogUhr_Zeiger_Stunden_Farbe_BlauAnteil,
    Vertikale_Verschiebung,Horizontale_Verschiebung,
    ZeigerLaenge,2 // Zeigerdicke ist 2
);

}

// ----- Uhr-Layout-Objekt: Zeigerursprung -----

function UhrLayout_ZeigerUrsprung_Erzeugen()
// Hinweis: Alle Zeiger sollten sich im Ursprung treffen: Beachte Dimension der Uhr !
//    Wenn sich alle Zeiger treffen, dann nach Augenmass alle Zeiger
//    in der Uhrfläche zentrieren.

```



```
// Wenn Zeiger zentriert wurden, dann Zeigerursprung als Ring darstellen und
// den Ursprung zentrieren.
{
// +++++ Form des Ursprunges

var ZeigerUrsprung_Flaeche=true; // false für Ringform
// Ringform geeignet zum Testen der Lage der Zeiger
// im Zeigerursprung (Zentrierung)

// Annahme: Ursprung als Ring
var UrsprungForm=2;
if (ZeigerUrsprung_Flaeche)
{UrsprungForm--;} // 1 für Fläche

// +++++ Dimension des Ursprunges

var Oval_BreiteUndHoehe=Math.floor(AnalogUhr_BreiteUndHoehe/12); // nicht verändern !!

// ----- Veränderung der Breite des Ursprunges
var BreiteVeraenderung = Math.abs(AnalogUhr_BreiteUndHoehe - 210);
BreiteVeraenderung= Math.floor(BreiteVeraenderung * 0.02);
BreiteVeraenderung*=-1;

// ----- Veränderung der Höhe des Ursprunges
var HoeheVeraenderung = Math.abs(AnalogUhr_BreiteUndHoehe - 210);
HoeheVeraenderung= Math.floor(HoeheVeraenderung * 0.08);
HoeheVeraenderung+=1;
HoeheVeraenderung*=-1;

// +++++ Verschiebung des Ursprunges

var Oval_Verschiebung_Horizontal=0;
var Oval_Verschiebung_Horizontal_Nenner=0;
var Oval_Verschiebung_Vertikal=0;
var Oval_Verschiebung_Vertikal_Nenner=0;

// ----- Verschiebung horizontal
// auf römisch prüfen
if (AnalogUhr_Ziffern_Satz==1)
{Oval_Verschiebung_Horizontal_Nenner=45;} // > 0 !!!
else
{Oval_Verschiebung_Horizontal_Nenner=25;} // > 0 !!!
Oval_Verschiebung_Horizontal=Math.floor(AnalogUhr_BreiteUndHoehe/Oval_Verschiebung_Horizontal_Nenner);
Oval_Verschiebung_Horizontal*=-1; // Verschiebung nach links

// ----- Verschiebung vertikal
// auf römisch prüfen
if (AnalogUhr_Ziffern_Satz==1)
{Oval_Verschiebung_Vertikal_Nenner=40;} // > 0 !!!
else
{Oval_Verschiebung_Vertikal_Nenner=30;} // > 0 !!!
Oval_Verschiebung_Vertikal=Math.floor(AnalogUhr_BreiteUndHoehe/Oval_Verschiebung_Vertikal_Nenner);
Oval_Verschiebung_Vertikal*=-1; // Verschiebung nach oben

// +++++ Objekt erzeugen

UhrLayout_ObjektKopf_Erzeugen('ID_AnalogUhr_Zeiger_Ursprung',BreiteVeraenderung,HoeheVeraenderung);

document.write(
    '<PARAM NAME="Line0001" VALUE="SetLineColor('
    + AnalogUhr_Zeiger_Ursprung_Farbe_RotAnteil + ','
    + AnalogUhr_Zeiger_Ursprung_Farbe_GruenAnteil + ','
    + AnalogUhr_Zeiger_Ursprung_Farbe_BlauAnteil
    + ') '
    + ' '
    + '>'
    + '<PARAM NAME="Line0001" VALUE="SetLineStyle(5)">'
    + '<PARAM NAME="Line0002" VALUE="SetFillStyle(' + UrsprungForm + ')">'
    + '<PARAM NAME="Line0003" VALUE="SetFillColor('
    + AnalogUhr_Zeiger_Ursprung_Farbe_RotAnteil + ','
    + AnalogUhr_Zeiger_Ursprung_Farbe_GruenAnteil + ','
    + AnalogUhr_Zeiger_Ursprung_Farbe_BlauAnteil
    + ') '
    + ' '
    + '>'
    + '>'

```




```

+ '<PARAM NAME="Line0004" VALUE="SetGradientFill(0,0,'
+     Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
+     Hintergrund_OvalVerschiebung_HorizontalUndVertikal + ','
+     '0)'
+     ""
+ '>'
+ '<PARAM NAME="Line0005" VALUE="Oval('
+     Oval_Verschiebung_Horizontal + ','
+     // Verschiebung Ovale horizontal bezüglich Zeigerursprung
+     // < 0 nach links, > 0 nach rechts
+     Oval_Verschiebung_Vertikal + ','
+     // Verschiebung vertikal bezüglich Zeigerursprung
+     // < 0 nach oben, > 0 nach unten
+     Oval_BreiteUndHoehe + ',' // Breite des Ovals
+     Oval_BreiteUndHoehe + ',' // Höhe des Ovals
+     '0)'
+     ""
+ '>'
+ '</OBJECT>'
+ );
}

// ##### Initialisierung der Uhr #####

function AnalogUhr_Init()
{
    var Div_BreiteUndHoehe=0;
    var Div_Left=0;

    // +++++ Maximale Höhe und Breite der Uhr prüfen, nicht ändern
    if (AnalogUhr_BreiteUndHoehe < 140)
    {AnalogUhr_BreiteUndHoehe=140;}

    if (AnalogUhr_BreiteUndHoehe > 330)
    {AnalogUhr_BreiteUndHoehe=330;}

    // +++++ ID der ActiveX-Controls festlegen, nicht ändern !!
    Objekt_ClassID_Uhrwerk= 'CLSID:B0A6BAE2-AAF0-11d0-A152-00A0C908DB96';
    Objekt_ClassID_UhrLayout='CLSID:369303C2-D7AC-11D0-89D5-00A0C90833E6';

    // +++++ Ovalverschiebung für Hintergrund-Objekt berechnen
    Hintergrund_OvalVerschiebung_HorizontalUndVertikal=Math.floor(AnalogUhr_BreiteUndHoehe/2); // nicht ändern
    Hintergrund_OvalVerschiebung_HorizontalUndVertikal*=-1; // nach links bzw. oben nicht ändern

    // +++++ Zeigerabweichung horizontal berechnen
    ZeigerAbweichung_Horizontal=Math.floor(AnalogUhr_BreiteUndHoehe / 42); // nicht ändern

    // +++++ Maximale Veränderung von Breite und Höhe definieren (siehe UhrLayout_ObjektKopf_Erzeugen() )
    Uhr_MaximaleVeränderung_BreiteUndHoehe=Math.floor(AnalogUhr_BreiteUndHoehe / 10); // nicht ändern

    // +++++ Div, der die gesamte Analoguhr umspannt
    //      Vorteil: Nur den Container verschieben und schon wandert die Uhr mitmit
    //      Die Uhr wird aus Objekten zusammengebaut, die in diesem Div liegen.

    // ----- Maximale Breite und Höhe
    Div_BreiteUndHoehe=AnalogUhr_BreiteUndHoehe + Uhr_MaximaleVeränderung_BreiteUndHoehe;

    // ----- Div-Left-Änderung ermitteln
    Div_Left=Math.floor(AnalogUhr_BreiteUndHoehe / 20); // nicht ändern
    Div_Left=AnalogUhr_Left - Div_Left;
    Div_Left+=3; // nicht ändern

    // ----- DIV-Kopf erzeugen
    document.write( '<DIV ID="ID_Div_AnalogUhr"'
+     ' STYLE="position:absolute;'
+     'width:' + Div_BreiteUndHoehe + ';'
+     'height:' + Div_BreiteUndHoehe + ';'
+     'top:' + AnalogUhr_Top + ';'
+     'left:' + Div_Left
+     ""
+ '>'
+ );

    // +++++ Das Uhrwerk erzeugen: Es ist der Motor der Uhr und wird NICHT gerendert

```



```
UhrWerk_Objekt_Erzeugen();

// +++++ Analoguhr aus Objekten zusammenbauen
//   Reihenfolge der Objekte entspricht der Überlagerung im Layout

// ----- Objekt des Hintergrundes der Analoguhr (nicht der Digitaluhr)
//   Wird der Hintergrund nicht erzeugt, so hat die Uhr das Ziffernblatt in der Farbe des normalen Hintergrundes
UhrLayout_Hintergrund_Erzeugen();

// ----- Objekt des Ziffernblattes erzeugen bei Bedarf
if (AnalogUhr_Ziffern_Erzeugen)
{UhrLayout_Ziffern_Erzeugen();}

// ----- Objekte der Zeiger
UhrLayout_AlleZeiger_Erzeugen();

// ----- Objekt des Zeigerursprunges
UhrLayout_ZeigerUsprung_Erzeugen();

// +++++ DIV-Ende
document.write( '</DIV>');
}
// -->
</SCRIPT>
</HEAD>

<BODY BGCOLOR="#3A6EA5">
<SCRIPT LANGUAGE="JScript">
<!--
// Es muss BODY angelegt sein: onload im BODY kommt zum Zeitpunkt, an dem der BODY noch nicht
// erzeugt wurde weil <BODY .....> noch nicht komplett geparkt wurde
//   Da init() auch document.write erzeugt, wird ein neuer weisser BODY erzeugt, der aber den
//   HEAD des alten Dokumentes nicht kennt und somit die Uhr zwar gerendert wird,
//   aber mangels der VBScripte NICHT startet.
AnalogUhr_Init();
// -->
</SCRIPT>
</BODY>
</HTML>
```



6. Anhang: Eigenschaften und Methoden des Internet Explorer

Hinweise

für ELEMENT das HTML-Tag des Objektes einsetzen

für object eine Referenz einsetzen z.B. laut ID-Attribut des Objektes (logischer Objektname), dessen Stringwert einen Zeigerbezeichner darstellt

Microsoft ändert fortlaufend die Active-X-Eigenschaften von Windows und somit auch des Internet Explorers

Diese fortlaufenden Änderungen muss der Programmierer in Erfahrung bringen.

Der Programmierer kann sich definitiv nicht auf Verfügbarkeit von Active-X-Controls verlassen und muss damit rechnen, dass seine Webseiten schlagartig nicht mehr komplett laufen weil u.a. Programmcode noch nicht angepasst ist. Ebenfalls muss der Programmierer Varianten von Windows und Patchzustände beachten, die prinzipiell Kostenprobleme verursachen können.

Mit anderen Worten: Wer Microsoft-Komponenten nutzt, muss wissen, was ihm blüht ... siehe nachfolgende Beispiel für Risiken.

Prinzipielle Lizenzprobleme für den Programmierer

Microsoft verlangt Lizenzierung von Windows. Bezüglich Windows-Versionen gibt es die Updatestufen z.B. per Servicepacks

Ein Windows mit Servicepack fällt unter die Lizenz des geupdateten Windows.

Ein Windows mit Vorversion zum Servicepack bedarf einer anderen Lizenz.

Will man z.B. den Internet Explorer 7 und 6 parallel testen, benötigt man 2 Windowslizenzen, da beide Versionen nicht parallel installierbar. Dazu kommt, dass es den IE 6 in 2 Versionen gibt: Win SP1 und SP2 (IE 7 nur ab Win SP2).

Für 3 Browserversionen benötigt man 3 Windowslizenzen, will man parallel testen.

Ein Blick auf Browser-Konkurrenzprodukte klärt die Sachlage unschlagbar: Opera ist z.B. parallel installierbar.

Hinweis: Man suche doch mal im Internet nach einem kostenlosen HTTP-Server vom Microsoft, um IE-Seite testen zu können, die JScript nutzen (inklusive Debugger). Denn sollte kein kostenloses Angebot findbar sein, kommen die Kosten von Entwicklungssoftware zum IE hinzu. Ein Blick auf Konkurrenz-HTTP-Server klärt die Sachlage: Apache-HTTP-Server ist kostenlos, allerdings nicht einfach einzurichten (Hinweis: Der HTTP-Server sollte virtuelle Hosts einrichten können und korrekt mit der Firewall des Users zusammenarbeiten können).

Abänderungen wegen Sicherheitspatches der jeweiligen Windows-Versionen

Abschaltungen von Active-X-Controls erfolgen auch im Rahmen der Sicherheitspatches zu Windows-Versionen.

Es ist auch möglich, dass wegen Sicherheitslücken abgeschaltet wird und somit Komponenten einer Webseite je nach Windowsversion nicht mehr laufen.

Im Rahmen der Sicherheitspatches ist es Microsoft sogar gelungen, Webseiten, die den MS-Encoder zur Komprimierung von

HTML- und JScript-Code nutzen, schlagartig unnutzbar zu machen: Ein Bug in einem Patch zu Windows XP - Q918899

Das Patch verursacht IE-Browser-Absturz bei per MS ScriptEncoder gepacktem JScript unter SP1 und 2 wenn HTTP 1.1 mit

Kompression genutzt wird z.B. bei

onclick-Handler auf IMG

klick ins Fenster per aktivem Popup

Der Absturz ist "read"-Fehler von immer ein und derselben Speicherstelle.

User, die dieses Patch installiert haben, können ab sofort keine IE-Seiten mit codiertem Script mehr ansehen.

Microsoft stellt Abhilfe nach geraumer Zeit zur Verfügung, jedoch spezifisch nach Windows XP-Version:

Patch Q918899 für

Windows XP SP1Download für jedermann bereitgestellt

SP2 nur auf kostenpflichtige telefonische Anfrage des Users per Downloadlink bereitgestellt, da

Microsoft explizit die User registriert haben will, bei denen das

Patchproblem auftritt (User muss sich Telefonnummer besorgen)

Solange also das Patch zum fehlerhaften Patch vom User nicht installiert wird,

z.B. weil der User keine Ahnung hat, dass und wo er sich die Telefonnummer

von Microsoft besorgen muss bzw. zu besorgen hat, wird der User

IE-Seiten mit komprimierten Code dauerhaft nicht nutzen können.

(Microsoft-Support ist z.T. nur in Englisch).

Abänderungen wegen Browser-Inkompatibilität

Popupblocker-Fehler

Die Microsoft Browser-Version IE 7 ist nicht abwärtskompatibel bezüglich Popup per window.createPopup()

Popup per window-Objekt ist ein Markenzeichen des IE, das im IE 7 nicht mehr fehlerfrei nutzbar ist.

Der Fehler liegt in der Popup-Blockerverwaltung des IE und wurde mit dem IE 7 implementiert.

Der Fehler tritt nicht auf, wenn ein Fenster per window.open() erzeugt wurde.

Bedingung:

Scriptfehleranzeige ist erlaubt im IE 7

Popupblocker ist im IE abgeschaltet

ein aktives Fenster (Register) mit Dokument, dass fortlaufend (rekursiv) genau 1 window.popup per .show() erzeugt.

ein weiteres Fenster (Register) z.B. leere Seite (about:blank)

beide (Register) liegen in einer gemeinsamen IE-Instanz

Ablauf: Wird Focus auf Register der leeren Seite gehalten und wird parallel das Popup per .show() erzeugt,



bricht der Browser das Dokument mit .show() ab (Scriptfehler).
 Der Popupblocker für die leere Seite verursacht den Programmfehler im Dokument mit .show(). Es wird folgende Meldung angezeigt (in der Informationsleiste):
 'Ein Popup wurde geblockt. Klicken Sie hier, um das Popup bzw. weitere Optionen anzuzeigen.'
 Die Bedeutung der Meldung laut Microsoft-Hilfe im IE 7:
 Der Popupblocker hat ein Pop-up-Fenster geblockt. Sie können den Popupblocker deaktivieren oder Popups temporär zulassen, indem Sie auf die Informationsleiste klicken.
 Die Realität zur obigen Meldung ist völlig anders:
 Linke oder rechte Maus auf die Meldung liefert z.B. Einstellungen darunter
 Popupblocker einschalten
 weitere Informationen
 jedoch keine Möglichkeit wie laut Bedeutung
 Damit gilt: Der abgeschaltete Popupblocker ist in Wirklichkeit aktiv.
 Pikant: Ein Popup erscheint normalerweise auch über fremde Fenster, die nicht das Popup erzeugt haben (z.B. Fenster einer Windowsanwendung z.B. einer anderen IE-Instanz)
 Der Popupblocker des IE bemerkt aber NUR Webseite, die das Popup erzeugt.
 Durch das Abwürgen von Popup wird das Popup natürlich auf und für anderen Seiten nicht relevant; im Falle einer anderen IE-Instanz also auch für diese nicht relevant, obwohl diese Instanz per Popupblocker verwaltet wird.
 Der Popupblocker beschneidet die Popup-Reichweite an der Wurzel, ist aber nicht objektorientiert zu den anderen Webseiten (die nicht das Popup erzeugt haben).
 Der Popupblocker ist nicht als Filter aufgesetzt sondern reingestrickt worden.
 Der Popupblockerfehler verändert die Eventverwaltung:
 Es werden u.a. ignoriert
 onfocus
 onblur
 onfocusin
 onfocusout
 und viele andere, so dass trotz Events z.B. des Body der Popupblockerfehler entsteht.

```
// nachfolgender Code setzt focus nicht neu: Fenstereintrag in Taskleiste blinkt eventuell
window.focus();
window.document.focus();
if(document.body!=null)
{if(document.body.style!='hidden')      // wenn hidden so focus() nicht möglich (Scriptfehler erzeugt)
 {document.body.focus();}
}
// wenn paralleles Fenster offen (on oder offline), so Scriptfehler erzeugt
popupzeiger.show(...);
```

Hinweis: Der Popupfehler ist so elementar, dass die vielen Beta-Testphasen des IE mehr als fragwürdig erscheinen, wie die Angabe von Microsoft, dass Code neu programmiert wurde, um den IE sicherer zu machen.

focus-Methode beim IE 7

windows.focus() document.focus() und body.focus() funktionieren NICHT
 zwischen Register in einem IE-Fenster
 zwischen Fensters z.B. in Taskleiste

Hinweis:

.focus() setzt Element aktiv, gibt dem Element den Focus und feuert dann onfocus
 .setActive() ist Teilmenge von .focus(): nur das aktiv setzen
 funktioniert nicht mit allen Elementen, mit denen .focus() funktioniert

animierte Gif (mit Timer)

Animierte Gifs (mit Timer), die unter IE 6 korrekt laufen, müssen unter IE 7 im Timer nicht mehr laufen:
 z.B. gar nicht mehr sichtbar, oder Timer nicht verwendet.
 Dann müssen animierte Gif-Bilder nach IE-Version bereitgestellt werden.

Abänderungen wegen Rechtstreitigkeiten von Microsoft mit Fremdanbietern

Ein sehr bekanntes Beispiel ist die nachträglich eingeführte Einschränkung von Active-X-Controls wegen Patentwahrung durch Microsoft, wobei für den JScript-Programmierer massive Änderungen eintreten.

Wegen Patentwahrung hat Microsoft ein zunächst freiwilliges Patch herausgegeben, dass bei ActiveX-Control per APPLET, EMBED oder OBJECT, die auf dem Bildschirm rendern (mit oder ohne Userschnittstelle), dafür sorgt, dass bei mouseover über das Control eine Sprechblase erscheint, die darauf hinweist, dass das Objekt als ActiveX-Control klickbar ist.
 Diese Sprechblase erscheint auch, wenn das Control keine Userschnittstelle hat, also diese gar nicht klickbar ist.

Es wurde das Eventmodell gleichzeitig geändert:

Es werden alle Events solange unterdrückt, bis der User die Sprechblase geklickt hat.
 Das Klicken muss auf das Objekt im Sprechblasenrahmen erfolgen, der so groß ist, wie die Dimension, in der gerendert wurde.



Es muss also ERST per Mausklick das Control aktiviert werden, ehe das Control klickbar und damit die Eventsteuerung aktiviert ist.
 Ein Control, dass programmtechnisch zwar was rendert, aber ansonsten ohne sichtbare programmtechnisch startet, muss ebenfalls geklickt werden, obwohl es bereits läuft und es nichts zu klicken gäbe (wenn keine Eventsteuerung eingebaut wurde).
 Wegen blockierter Eventsteuerung ist also die Sprechblase z.B. nicht automatisch klickbar.
 Die Eventauslösung per nicht-objekteigenen Eventhandler, der für das Objekt per fireEvent() ein Event auslöst, ist solange blockiert, bis der User die Sprechblase geklickt hat.

style.visibility='hidden' wird ignoriert

Die Sprechblase erscheint auch dann, wenn das Control mit style.visibility='hidden' belegt ist, also sich unsichtbar rendert:
 Der Sprechblasenrahmen hat genau die Dimension wie die des unsichtbaren Controls. Der Sprechblasenrahmen erscheint also Zusammenhangslos, und der User weiß nicht, warum er klicken soll, wenn er nichts sieht. Vor allem weiß er nicht, WAS er klickt ... ideale Basis für Schadsoftware per Script.

Diese Sprechblase erscheint nur DANN NICHT, wenn die Userschnittstelle mit Breite == Höhe == 0 gerendert wird. Sollte die Userschnittstelle in einem Container liegen, z.B. DIV, dann wird der Container, wenn er in der Dimension kleiner ist, also die Userschnittstelle, angepasst. Daher muss der Container ebenfalls mit Breite == Höhe == 0 gerendert werden. Wegen Dimensionierung auf 0 sollte style.visibility="hidden" sein. Im Falle eines Containers reicht es, den style des Containers zu ändern, da visibility normalerweise vererbt wird an Kinder, also auch an das Control.

Abänderung wegen Abschaltungen

DirectX ist wegen Abschaltung von Active-X--Controls nicht mehr abwärtskompatibel:

Z.B. wurde bei Win XP SP2 Direct Animation aus DirectX schlagartig durch Abschaltung von Bibliotheken dezimiert, die es bei Win XP SP1 aber noch gibt.

Hier ein Beispiel aus dem Jahr 2004: Abschaltungen von Active-X-Controls

ActiveX-Controls und Unterstützung/Verbot 20041215

erlaubt sind noch

Tabular Data-Steuerelement{333C7BC4-460F-11D0-BC04-0080C7055A83} Das TDC (Tabular Data-Steuerelement) ermöglicht die Weiterverarbeitung von Daten, die nur im Textformat vorliegen, beispielsweise durch Darstellung in einer Tabelle oder Sortierung. Weitere Informationen:•

http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp(http://msdn.microsoft.com/workshop/database/tdc/tabular_data_control_node_entry.asp)

Microsoft Agent Control - Version 2.0 {D45FD31B-5C6E-11D1-9EC1-00C04FD7081F} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Weitere Informationen:•

<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>(<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>)

Microsoft MSChat-Steuerelement-Objekt 2.0 - 2.5 {D6526FE0-E651-11CF-99CB-00C04FD64497}

Dieses Steuerelement wird von Webautoren verwendet, um text- und graphisch basierte Chatgemeinden für Echtzeitkonversationen im Web zu erstellen.

Microsoft ActiveX Upload-Steuerelement, Version 1.5 {886e7bf0-c867-11cf-b1ae-00aa00a3f2c3} Dieses Steuerelement kann auf vielerlei Art genutzt werden, um auf einfache Weise Webinhalte via Drag and Drop zu veröffentlichen. Weitere Informationen:• 230298 (<http://support.microsoft.com/kb/230298/DE/>) - Posting Acceptor Release Notes

• http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp
 (http://msdn.microsoft.com/workshop/management/tools/reference/file_upload_control.asp)

verboten sind



Datenbindung RDS {BD96C556-65A3-11D0-983A-00C04FC29E36} {BD96C556-65A3-11D0-983A-00C04FC29E33} Die RDS (Remote Data Service) Steuerelemente ermöglichen dem Browser, client-basierte SQL Abfragen an einen Webserver zu stellen. Inzwischen wurde RDS jedoch durch neuere Standards wie SOAP abgelöst, von einer weiteren Verwendung von RDS wird daher abgeraten. Weitere Informationen:• 184375 (<http://support.microsoft.com/kb/184375/DE/>) - Sicherheitsaspekte bei RDS 1.5, IIS 3.0 oder 4.0 und ODBC

<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>
(<http://msdn.microsoft.com/library/en-us/iissdk/iis/remotedatabindingwithremotedataservice.asp>)
http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp
(http://msdn.microsoft.com/library/en-us/dnmdac/html/data_mdacroadmap.asp)

XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer {550dda30-0541-11d2-9ca9-0060b0ec3d39} {CFC399AF-D876-11d0-9C10-00C04FC99C8E} {e54941b2-7756-11d1-bc2a-00c04fb925f3} {7108ECB4-AFDC-11D1-ADC1-00805FC752D8} XMLDSO, XMLDocument, DOMDocument, und XMLIslandPeer ermöglichen die Verarbeitung von XML Daten, etwa die Bindung von HTML Elementen an einen XML Datensatz, oder das Einlesen, Manipulieren, und Zurückschreiben von XML Daten.

Die Steuerelemente DOMDocument und XMLIslandPeer bzw. die dazugehörigen ClassIDs sind nicht mehr aktuell, so dass von einer generellen Freigabe dieser Steuerelementgruppe abgeraten wird. Weitere Informationen:• http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp(http://msdn.microsoft.com/library/en-us/xmlsdk/htm/xml_concepts2_7ook.asp)

Internet Explorer

Active Setup / IE Active Setup-Steuerelement {F72A7B0E-0DD8-11D1-BD6E-00AA00B92AF1} Dieses Steuerelement enthält die in Microsoft Security Bulletin MS99-037 beschriebene Sicherheitsanfälligkeit. Um eine weitere Ausführung zu verhindern wurde im Rahmen dieses Security Bulletins ein Kill-Bit gesetzt, so dass selbst bei einer Freigabe dieses Controls eine Ausführung blockiert wird. Weitere Informationen:• <http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>(<http://www.microsoft.com/technet/security/bulletin/ms99-037.msp>)
<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>
(<http://www.microsoft.com/technet/security/bulletin/fq99-037.msp>)
240797 (<http://support.microsoft.com/kb/240797/DE/>) - So verhindern Sie die Ausführung von ActiveX-Steuerelementen in Internet Explorer

Media Player / Active Movie Runtime {A4001DE0-7075-11d0-89AB-00A0C9054129} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Microsoft NetShow Player {2179C5D3-EBFF-11CF-B6FD-00AA00B4E220} Die Funktionalität dieses Steuerelements wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das NetShow Player Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wird abgeraten.

Media Player / Windows Media Player {22D6F312-B0F6-11D0-94AB-0080C74C7E95} Dies ist das Steuerelement für Windows Media Player version 6.4 und war Installationsbestandteil bis einschließlich Windows Media Player Version 8. Ab Windows Media Player 9 wurde diese ClassID durch die neue ClassID {6BF52A52-394A-11D3-B153-00C04F79FAA6} abgelöst, deren Verwendung stattdessen empfohlen wird. Ab Windows Media Player Version 9 wird ferner die alte ClassID anhand eines Wrappers automatisch auf die neue



ClassID umgeleitet. Die ClassID für Windows Media Player Version 9 ist jedoch nicht in der Liste der vom Administrator genehmigten Steuerelemente enthalten, und muss bei Bedarf manuell hinzugefügt werden.

Animierte Schaltflächen {0482B100-739C-11CF-A3A9-00A0C9034920} Dieses Steuerelement erlaubte in frühen Versionen des Internet Explorer die Verwendung animierter Schaltflächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von der Freigabe des Steuerelements wird daher abgeraten.

IE Label-Steuerelement

{99B42120-6EC7-11CF-A6C7-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 auch kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

IE Menu-Steuerelement {74701400-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ermöglicht die Handhabung von Menüstrukturen in Webseiten, wird jedoch nicht mehr unterstützt und dürfte nur noch selten Verwendung finden. Von einer Freigabe des Steuerelements wird daher abgeraten.

IE Preloader-Steuerelement {16E349E0-702C-11CF-A3A9-00A0C9034920} Dieses Steuerelement ermöglichte das Vorladen von Webseiten, ist jedoch inzwischen nicht mehr aktuell, wird nicht mehr unterstützt und dürfte nicht mehr im Einsatz sein. Aufgrund einer potentiellen Sicherheitsanfälligkeit in diesem Steuerelement wird von einer Freigabe abgeraten. Weitere Informationen: • 231452 (<http://support.microsoft.com/kb/231452/DE/>) - Update Available for "Legacy ActiveX Control" Issue

IE Timer-Steuerelement {59CCB4A0-727D-11CF-AC36-00AA00A47DD2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

MCSiMenü {275E2FE0-7486-11D0-89D6-00A0C90C9B67} Dieses Steuerelement dient der Anpassung von Pop-upmenüs, ist jedoch nicht mehr aktuell und wurde nach Windows 98 nicht mehr ausgeliefert. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten.

Pop-upmenüobjekt {7823A620-9DD9-11CF-A662-00AA00C066D2} Dieses Steuerelement ist nicht mehr aktuell und seit Internet Explorer Version 5 kein Bestandteil der Installation mehr. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: • 190045 (<http://support.microsoft.com/kb/190045/DE/>) - INFO: ActiveX Controls That Are Removed from Internet Explorer 5

Microsoft Agent Control - Version 1.5 {F5BE8BD2-7DE6-11D0-91FE-00C04FD701A5} Microsoft Agent repräsentiert die neue Generation des ursprünglichen Office-Assistenten. Anstatt den Assistenten jedoch innerhalb eines Rahmens darzustellen wird hier lediglich der Charakter bzw. Agent selbst dargestellt und kann auch in Webseiten verwendet werden. Diese Version des Steuerelements ist jedoch nicht mehr aktuell und wird nicht mehr unterstützt. Von einer Freigabe des Steuerelements wird daher abgeraten. Weitere Informationen: •



<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>
<http://msdn.microsoft.com/library/partbook/egvb6/introducingmicrosoftagent.htm>

Eigenschaften

@page

Regel für Dimensionen, Orientierungen und Margins in einer Seite per styleSheet Objekt

Regel ist eine Pseudoklasse von Style

ab IE 5.5

Seite wird als Rechteckbereich aufgefasst, der eingeteilt ist in:

Seitenbereich: Inhalt der Seite z.B. Text, Grafik

Marginbereich: Rand um den Seitenbereich

siehe auch Objekt page

":first" Regel gehört der 1. Seite

":footer" Regel gehört der Fußnote

":header" Regel gehört der Kopfnote

":left" Regel gehört der linken Seite

":left : header" Regel gehört der Kopfnote Seite links

":left : footer" Regel gehört der Fußnote Seite links

":right" Regel gehört der rechten Seite

":right:header" Regel gehört der Kopfnote Seite rechts

":right:footer" Regel gehört der Fußnote Seite rechts

Beispiel: @page : left { font-weight:bold;font_style:italic; }

.0 bis .n

Wert des Argumentes mit Index 0 ... bzw. n im Script-Objekt arguments als Eigenschaft eines

Funktionsobjektes (siehe Script-Objekt Function und Anweisung function)

.0 entspricht auch funktions_bezeichner.arguments[0]

.n entspricht auch funktions_bezeichner.arguments[n]

n von 0 bis beliebig ganzzahlig-numerische Ziffernfolge ohne Vornull

ist die Position ab 0 innerhalb der Argumentenliste von links nach rechts

Parameterliste von links nach rechts

Hinweis: Argumentenliste und Parameterliste mit gleicher Elementanzahl, denn

jedes Argument muss mit Wert versorgt werden

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

var GlobalesFeld = new Array();

function FunktionsBezeichner()

{

// Argumentenliste der Funktion **lokal** zur Funktion referenzieren

var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

// Anzahl der Argumente

var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

if (ArgumenteAnzahl > 0)

{

// Arrgumentenliste auslesen

GlobalesFeld[0] = ArgumentenListeAlsFeld.0;

GlobalesFeld[1] = ArgumentenListeAlsFeld.1;

}

// hier die weiteren Anweisungen der Funktion

}

.abstract

Beschreibung einer Media-Datei auf der Timeline

bei Advanced Stream Redirector (ASX)-Datei wird der Text unter ABSTRACT

vom aktuellen Eintrag geliefert und nicht der Datei selbst

siehe Objekt currTimeState und Behavior .style.time2

Beispiel 1:

<HTML XMLNS:t="urn:schemas-microsoft-com:time">

<HEAD>

<?IMPORT namespace="t" implementation="#default#time2">

<STYLE>

.time_line_klasse { behavior: url(#default#time2) }

</STYLE>

</HEAD>

<BODY>

<t:VIDEO

ID="ID_Video"

SRC="test.wmv"

STYLE="position:absolute;top:50px;height:100px"

>



```

</t:VIDEO>
<SPAN ID="ID_Span"
      STYLE="position:absolute;top:165px;"
>
</SPAN>
<BUTTON ID="ID_Button"
        onclick="ID_Span.innerText= ID_Video.abstract"
>
      Klick
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
      .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
      function ButtonUpdate()
      {
          if (ID_Media.currTimeState.isActive)
          {
              ID_Button1.disabled=true;
              ID_Button2.disabled=false;
              ID_Button3.disabled=false;
              ID_Button4.disabled=false;
          }
          else
          {
              ID_Button1.disabled=false;
              ID_Button2.disabled=true;
              ID_Button3.disabled=true;
              ID_Button4.disabled=true;
          }
      }

      function AnzeigeUpdate()
      {
          ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
          ID_Span3.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
          ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
          ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
          ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
          ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
          ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
          ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
      }

      function AnzeigeLoeschen()
      {
          ID_Span1.innerText = "Titel: ";
          ID_Span2.innerText = "Autor: ";
          ID_Span3.innerText = "Abstract: ";
          ID_Span4.innerText = "Copyright: ";
          ID_Span5.innerText = "Filename: ";
          ID_Span6.innerText = "Banner: ";
          ID_Span7.innerText = "Banner Abstract: ";
          ID_Span8.innerText = "Banner MoreInfo: ";
      }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

      <t:MEDIA ID="ID_Media"
              SRC="test.asx"
              BEGIN="indefinite"
              TIMEACTION="visibility"
              onend="ButtonUpdate();"
              ontrackchange="AnzeigeUpdate();"
              onmediacomplete="ButtonUpdate();AnzeigeUpdate();"

```



```

>
</t:MEDIA>

<SPAN ID="ID_Span1">Titel:</SPAN>
<BR>
<SPAN ID="ID_Span2">Autor:</SPAN>
<BR>
<SPAN ID="ID_Span3">Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span4">Copyright:</SPAN>
<BR>
<SPAN ID="ID_Span5">Filename:</SPAN>
<BR>
<SPAN ID="ID_Span6">Banner:</SPAN>
<BR>
<SPAN ID="ID_Span7">Banner Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

<BR>
<BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
>
    Start
</BUTTON>
<BUTTON ID="ID_Button2"
        onclick="ID_Media.playlist.nextTrack();"
>
    naechster Track
</BUTTON>
<BUTTON ID="ID_Button3"
        onclick="ID_Media.playlist.prevTrack();"
>
    vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"
>
    Stop
</BUTTON>
</BODY>
</HTML>

```

.accelerate Beschleunigung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute `.repeatCount` oder `.repeatDur`
 Summe der Werte der Attribute `.accelerate` und `.decelerate` darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
 siehe Objekt `currentTimeState` und Behavior `.style.time2`
 siehe `.decelerate`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:ANIMATE TARGETELEMENT="ID_Div"
              ATTRIBUTENAME="left"
              TO="400"
              DUR="3"
              ACCELERATE="1"
              REPEATCOUNT="3"
    >
    </t:ANIMATE>
    <DIV ID="ID_Div" CLASS="time_line_klasse" STYLE="position: absolute;left:10px">
    </DIV>
</BODY>
</HTML>

```

.accept Liste von Multipurpose Internet Mail Extensions (MIME)-Typen
 z.B. "text/html"



```
"image/png"
"image/gif"
"video/mpeg"
"audio/basic"
"text/tcl"
"text/javascript"
"text/vbscript"
```

Listenelemente mit Komma trennen

Beispiel:

```
<INPUT TYPE=file ACCEPT="text/html, image/png">
```

es können nur laut ACCEPT-Wert angegebene Dateitypen selektiert werden

.acceptCharset

Liste von UTF-8 Zeichen für Encoden der Eingabedaten durch den Server
Liste deklariert alle Zeichen, die nicht im Charset des Dokumentes erfasst werden
Liste wird vom Server benötigt
wenn nicht kodiert, so nur der Charset des Dokumentes verwendet
UTF-8 Zeichen: Teil des Unicode (0 bis 255)

.accessKey

Tastaturzugriff auf ein Objekt per Alt + Taste
bei Ausführung der Tastenkombination wird
das Sprungziel wird focussiert
die Sprungquelle defocussiert
das Focus-Ereignis ausgelöst
vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt

Beispiel:

```
<LABEL FOR="ID_Input" ACCESSKEY="I">
  #<SPAN>1</SPAN>:
  Alt+I fuer Sprung zur Textbox
</LABEL>
<INPUT TYPE="text"
  ID="ID_Input "
  NAME="T1"
  VALUE="text1
  SIZE="20"
  TABINDEX="1"
  >
```

.accumulate

Kumulative Animation eines Elementes auf der Timeline
Element darf nicht aktiv sein, wenn .accumulate definiert werden soll:
Element erst danach starten.
Es kann jede numerische Style-Eigenschaft kumulativ verändert werden und damit
die kumulative Animation des Elementes in der Style-Eigenschaft erzeugen.
Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des
Wertes sein (mathematische Berechnungen erst nach Entfernung der Einheit
per Stringoperationen möglich)
Name der Eigenschaft laut .attributName
Änderung des Wertes pro Durchlauf bzw. kumulativ um den Maximalwert laut .by
Anzahl der Durchläufe bzw. der Wert-Kumulationen laut .repeatCount
Kumulationsschrittweite laut .calcMode
Ereignis des Startes eines Durchlaufes bzw. Kumulation um Maximalwert ist onrepeat.
Wert der Eigenschaft .repeatCount muss > 1 sein
Kumulation nicht aktiv:
Nach jedem Durchlauf laut .repeatCount erfolgt Rücksetzen der Style-Eigenschaft des
Elementes auf den Zustand vor dem Durchlaufbeginn.
Es wird also die Style-Eigenschaft nicht wertmäßig kumuliert.
Das Element wird also laut .repeatCount mehrmals animiert.
Pro Durchlauf wird für die Eigenschaft laut .attributName der Wert
ab Startwert
in Schrittweite laut .calcMode
um den maximalen Wert laut .by
erhöht.
Kumulation aktiv:
Die Style-Eigenschaft des Elementes wird pro Durchlauf wertmäßig laut
.repeatCount um den Wert laut .by kumuliert.
Kumulationsschrittweite laut .calcMode
Das Element wird also genau 1 mal animiert.
Mit Beginn des Startes des Elementes wird für die Eigenschaft laut .attributName
der Wert ab Startwert
in Schrittweite laut .calcMode
um den maximalen Wert aus dem Produkt aus
.by mal Anzahl der Wiederholungen
z.B. laut .repeatCount
erhöht.



Es wird also über **alle** Wiederholungen der Wert kumuliert.
 siehe `.additive` `.calcMode`
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft `.width`) nach rechts:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function SpanInhaltInit(Kette)
    {
        // Zustand
        ID_Span1.innerText = "";

        // Kumulationsart
        ID_Span2.innerText =Kette;

        // Zaehler auf 1
        ID_Span3.innerText = "1";

        // DIV-Text festlegen
        ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerText += " genau 1x kumulativ um ";
            ID_Div.innerText += ID_Animate.by
            ID_Div.innerText += " mal "
            ID_Div.innerText += ID_Animate.repeatCount;
        }
        else
        {
            // keine Wertkumulation von .width
            ID_Div.innerText += ID_Animate.repeatCount;
            ID_Div.innerText += " mal um ";
            ID_Div.innerText += ID_Animate.by
        }

        ID_Div.innerText += " aus";
    }

    function KumulationAus()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("none");
        ID_Animate.accumulate="none";

        // DANACH Animation starten
        ID_Animate.beginElement();
    }

    function KumulationEin()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("sum");
        ID_Animate.accumulate="sum";

        // DANACH Animation starten
        ID_Animate.beginElement();
    }

    function Anzeige()
    {
        ID_Span1.innerText = "Animation ist beendet ";
    }
  </SCRIPT>
</HTML>
```



```

</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE      ID="ID_Animate"
                   TARGETELEMENT="ID_Div"
                   ATTRIBUTENAME="width"
                   BY="150px"
                   DUR="3"
                   REPEATCOUNT="3"
                   BEGIN="indefinite"
                   FILL="freeze"
                   onend="Anzeige();"
                   onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    </t:ANIMATE>

    animierter DIV
    <DIV      ID="ID_Div"
             CLASS="time_line_klasse"
             STYLE= "position:absolute;
                    top:125px;
                    left:25px;
                    height:100px;
                    width:125px;
                    border:solid black 2px;
                    "
    >
    </DIV>
    <BR>

    Zustand der Animation:
    <SPAN ID="ID_Span1"></SPAN>
    <BR>

    Kumulationsart:
    <SPAN ID="ID_Span2"></SPAN>
    <BR>

    Durchlaufzaehler:
    <SPAN ID="ID_Span3" ></SPAN>
    <BR>

    <BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
    <BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.action Url des Servers und des dortigen Dokumentes bei Formular

Beispiel 1:

```

<HTML>
    <FORM ACTION="http://www.test.de/sample.asp" METHOD="POST">
        <SELECT NAME="Flavor">
            <OPTION VALUE="Test1"> Test1
            <OPTION VALUE="Test2"> Test2
            <OPTION VALUE=" Test3" SELECTED> Test3
        </SELECT>
        <INPUT TYPE=SUBMIT>
    </FORM>
</HTML>

```

Beispiel 2:

```

<FORM ACTION="mailto:test@test.de" METHOD=GET>
    <INPUT  NAME=subject TYPE=hidden
        VALUE="Testt%20Product%20Information%20Anforderung"
    >
    volle Mailadresse eingeben
    <BR>
    <TEXTAREA NAME=body COLS=40></TEXTAREA>
    <INPUT TYPE=submit VALUE="absenden "
</FORM>

```

Beispiel 3:

```

<FORM ACTION="test.htm">
    <INPUT TYPE="submit" VALUE="Gehe zu test.htm">

```



</FORM>

Beispiele für mailto-Formen:

Standard	mailto:aaa@bbb
carbon copy	mailto:aaa@bbb?cc=mailto:ccc@ddd (mit Kopie an anderen Empfänger)
blind copy	mailto:aaa@bbb?bcc=mailto:ccc@ddd (mit Blind-Kopie an anderen Empfänger)
carbon copy und Betreff	mailto:aaa@bbb?cc=mailto:ccc@ddd&subject=betreff_text
carbon copy und Betreff und Mailtext	mailto:aaa@bbb?cc=mailto:ccc@ddd&subject=betreff_text&body=mail_text

Betreff/Mailtext analog für Standard und blind copy

Beispiel für Spam-Schutz einer Email-Adresse (Schutz vor Missbrauch nach dem Scannen der Email-Adresse durch Suchmaschine):

// test@test.de wird per Script und nicht als HTML im Quellcode kodiert

```

var user_name="test";
var host_name="test.de";

document.write(      '<A HREF="mailto:'
                    + user_name
                    + '@'
                    + host_name
                    + '>'
                    + user_name
                    + '@'
                    + host_name
                    + '</A>'
                    );

```

Beispiel für Festplattenverzeichnis unter Windows auslesen:

```

<BODY>
<FORM ACTION="file:///C:/\"
      METHOD="GET"
>
      <INPUT TYPE="submit" VALUE="Root von LW C anzeigen!">
</FORM>
</BODY>

```

.activeDur totale Dauer der Timeline in Sekunden
 inklusive aller Wiederholungen und Autoreverse
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
      .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
      function DauerAnzeigen()
      {alert('Totale Dauer = ' + ID_Timeline.currTimeState.activeDur + ' Sekunden');}
</SCRIPT>
</HEAD>
<BODY>
      <!-- 3 * 3 * 2 = 18 Sekunden-->
      <t:EXCL ID="ID_Timeline"
            DUR="3"
            REPEATCOUNT="3"
            AUTOREVERSE="true"
      >
            <DIV ID="ID_Div"
                  CLASS="time_line_klasse"
                  BEGIN="0"
                  DUR="1"
            >
                  1
            </DIV>
      </t:EXCL>

```




```

<BR>
<BUTTON          ID="ID_Button"
                 onclick=" DauerAnzeigen()"
>
        Klick mich
</BUTTON>
</BODY>
</HTML>

```

.activeElement Referenz auf dasjenige Objekt im Dokument , das Focus hat
 IE unter 5.x Zeiger auf BODY-Objekt
 ab IE 5.x null

Beispiel:

```
var Zeiger = document.activeElement
```

.activeTime aktueller Zeitpunkt in Sekunden in der Timeline ab Start
 inklusive aller Repeats, autoReverse
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
        .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
        function WindowOnLoad()
        {
                // Timer mit Intervall von 100 Millisekunden
                window.setInterval(Anzeigen, 100);
        }

        function Anzeigen()
        {
                ID_Div2.innerHTML = "simpleTime:&nbsp;"
                                + (ID_Animation.currTimeState.simpleTime);

                ID_Div3.innerHTML = "segmentTime:&nbsp;"
                                + (ID_Animation.currTimeState.segmentTime);

                ID_Div4.innerHTML = "activeTime:&nbsp;"
                                + (ID_Animation.currTimeState.activeTime);
        }

        function Wiederholen()
        {
                ID_Div1.innerHTML = "repeatCount:&nbsp;"
                                + (ID_Animation.currTimeState.repeatCount + 1 );
                                // repeatCount ab 0, aber Anzeige ab 1
        }
        window.onload = WindowOnLoad;           // mit () so sofort ausgeführt
                                                // ohne () so reiner Zeiger

</SCRIPT>
</HEAD>
<BODY>
        <DIV>
                <DIV ID="ID_Div1">repeatCount:&nbsp;1</DIV>
                <DIV ID="ID_Div2">simpleTime:&nbsp;0</DIV>
                <DIV ID="ID_Div3">segmentTime:&nbsp;0</DIV>
                <DIV ID="ID_Div4">activeTime:&nbsp;0</DIV>
        </DIV>

        <DIV ID="ID_DivTimeLine"
                CLASS="time"
                STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
        >
        </DIV>
        <t:ANIMATEMOTION          ID="ID_Animation"
                                TARGETELEMENT="ID_DivTimeLine"
                                TO="340,40"
                                DUR="2"
                                AUTOREVERSE="true"
                                REPEATCOUNT="5"

```



```

                                FILL="freeze"
                                onrepeat="Wiederholen()"
                            >
                        </t:ANIMATEMOTION>
    </BODY>
</HTML>

```

.activeTrack Referenz auf aktives Objekt `playItem`, das gerade in der Playliste aktiv ist
 Achtung: Playliste muss aktiv sein, sonst wird Fehler erzeugt
 Aktivstatus prüfen per Eigenschaft `.isActive`
 Track entspricht `playItem` Objekt
 Trackliste liegt in der Behavior-Collection `.style.time2.playList` als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
 aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
 aktiver Track in der Liste: wird abgespielt
 Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten

Beispiel 1:

```

object.playList.activeTrack = object.playList.item(index);
var Zeiger = object.playList.aktiveTrack;

                                index      Integer, ab 0

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playList.activeTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playList.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playList.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playList.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playList.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playList.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playList.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playList.activeTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
    }
</SCRIPT>

```



```

</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA          ID="ID_Media"
                      SRC="test.asx"
                      BEGIN="indefinite"
                      TIMEACTION="visibility"
                      onend="ButtonUpdate();"
                      ontrackchange="AnzeigeUpdate();"
                      onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON          ID="ID_Button1"
                    onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON          ID="ID_Button2"
                    onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON          ID="ID_Button3"
                    onclick="ID_Media.playlist.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON          ID="ID_Button4"
                    onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>
</HTML>

```

.additive numerische Werte von Eigenschaften eines Elementes ersetzen oder zu den numerischen Werten der gleichnamigen Eigenschaften anderer Elemente addieren während der Elemente-Animation auf der Timeline
Wert der Eigenschaft: numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein gültige Eigenschaften sind z.B. `.values` oder `.by`
sinnvoll für Kombinationen von Elementen (auch bei Wiederholung eines Elementes)
Element darf nicht aktiv sein, wenn `.additive` definiert werden soll:
Element erst danach starten.
Achtung: Im Objekt `animatecolor` (`t:ANIMATECOLOR`) ist `.values` auch als Farbliste definiert !
siehe `.accumulate` `.by` `.attributeName`
siehe Objekt `currTimeState` und Behavior `.style.time2`

.align Ausrichtung
"center" default nur bei TH-Tag
"justify"
"left" default außer bei TH-Tag
"right"

Beispiel für Erzeugung einer Tabelle in Script per HTML, TOM und DOM:

```
<HTML>
```



```

<HEAD>
<SCRIPT LANGUAGE="JScript">
    function TabelleFuellen()
    {
        var Zeile;
        var Zelle;

        //+++++ Tabellenrahmen und Rahmenfarbe ++++++
        ID_Tabelle.border = 1;
        ID_Tabelle.bgColor = "magenta";

        //+++++ TabellenKopf füllen ++++++
        //----- erzeugen
        var TabellenKopf = ID_Tabelle.createTHead();

        //----- Hintergrundfarbe
        TabellenKopf.bgColor = "blue";

        //----- mit 1 Zeile
        Zeile = TabellenKopf.insertRow();

        // Zeile mit 1 Zellenfüllen
        Zelle = Zeile.insertCell();
        Zelle.align = "center";
        Zelle.style.fontWeight = "bold";
        Zelle.innerText = "Tabellenkopf Zelle";

        //+++++ Tabellenbody 0 füllen ++++++
        //----- wurde per HTML-Kodierung im BODY erzeugt
        //----- Hintergrundfarbe
        ID_TBody0.bgColor = "green";

        //----- 2 Zeilen mit je 1 Zelle erzeugen
        for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
        {
            //----- Zeile erzeugen
            Zeile = ID_TBody0.insertRow();

            //----- Zelle in der Zeile erzeugen
            Zelle = Zeile.insertCell();

            // und füllen
            Zelle.innerText = "TBODY 0 Zeile " + ZeilenZahler + " Zelle"
        }

        //+++++ Tabellenbody 1 füllen ++++++
        //----- wurde per HTML-Kodierung im BODY erzeugt
        //----- Hintergrundfarbe
        ID_TBody1.bgColor = "yellow";

        //----- 2 Zeilen mit je 1 Zelle erzeugen
        for ( var ZeilenZahler =0 ; ZeilenZahler <2; ZeilenZahler ++ )
        {
            //----- Zeile erzeugen
            Zeile = ID_TBody1.insertRow();

            //----- Zelle in der Zeile erzeugen
            Zelle = Zeile.insertCell();

            // und füllen
            Zelle.innerText = "TBODY 1 Zeile " + ZeilenZahler + " Zelle"
        }

        //+++++ TabellenFuss füllen ++++++
        //----- erzeugen
        var TabellenFuss = ID_Tabelle.createTFoot();

        //----- mit Hintergrundfarbe
        TabellenFuss.bgColor = "brown";

        //----- mit 1 Zeile
        Zeile = TabellenFuss.insertRow();

        // Zeile mit 1 Zellenfüllen

```



```

        Zelle                = Zeile.insertCell();
        Zelle.align          = "center";
        Zelle.style.fontWeight = "bold";
        Zelle.innerText      = "Tabellenfuss Zelle";
        Zelle.colSpan        = "1";
        Zelle.id              = "ZelleImTabellenFuss";

        // ++++ TabellenCaption füllen +++++
        // ----- erzeugen
        var TabellenCaption = ID_Tabelle.createCaption();

        // ----- und füllen
        TabellenCaption.align = "bottom";
        TabellenCaption.style.fontSize = "10";
        TabellenCaption.innerText = "TabelleCaption"
    }

    function ZelleImTabellenFussAendern()
    { ZelleImTabellenFuss.innerText = "Tabellenfuss Zelle neu" }
</SCRIPT>
</HEAD>
<BODY onload="TabelleFuellen();">
    <!-- Tabelle leer mit allen TBODY erzeugen, aber komplett ohne Daten, da sonst nicht manipulierbar !!! /-->
    <TABLE ID="ID_Tabelle"
        BORDER
        BGCOLOR="gray"
    >
        <TBODY ID="ID_TBody0"></TBODY>
        <TBODY ID="ID_TBody1"></TBODY>
    </TABLE>
    <BR>
    Tabelle mit HTML und TOM erzeugt
    <BR>
    <BUTTON onclick="ZelleImTabellenFussAendern();">Zelle im Tabellenfuss aendern</BUTTON>
</BODY>
</HTML>

```

.aLink Farbe eines Alinks im Dokument
 document.body.aLink
 "#rrggbb" oder vordefinierte Farbname

.alinkColor Farbe aller aktiven Links im Dokument
 document.body.alinkColor
 "#rrggbb" oder vordefinierte Farbname

.allowTransparency Transparenz ein/aus
 Transparenz: z.B. Hintergrundfarbe des Elternobjektes wird zur Hintergrundfarbe des Frames
 Achtung: Wenn Frame-eigenes STYLE-Attribut mit Wert für background-color
 also STYLE="background-color: farb_bezeichner"
 kodiert wurde, so wird die Transparenz ignoriert und die
 Hintergrundfarbe laut STYLE verwendet

Beispiel:

```

<BODY STYLE="background-color: red">

    // roter Hintgrund im Frame laut Eltern = Body
    <IFRAME ID="Frame1" SRC="transparentBody.htm" allowTransparency="true"></IFRAME>

    // roter Hintgrund im Frame laut Eltern wird durch STYLE-Wert überschrieben
    //                      also grün angezeigt
    <IFRAME ID="Frame2" SRC="transparentBody.htm" allowTransparency="true"
        STYLE="background-color: green">
    </IFRAME>

    // roter Hintgrund im Frame laut Eltern nicht angezeigt, sondern Standard = weiss
    <IFRAME ID="Frame3" SRC="transparentBody.htm"></IFRAME>

    // roter Hintgrund im Frame laut Eltern nicht angezeigt, auch nicht Standard = weiss
    //                      sondern laut STYLE-Wert, also grün
    <IFRAME ID="Frame4" SRC="transparentBody.htm" STYLE="background-color: green">
    </IFRAME>
</BODY>

```

.alt Alternativer Text als Tooltip
 Beispiel:



```
<IMG SRC="http://www.test.de/test.gif" ALT="Ein Testbild ">
```

.altHTML HTML-Script, der ausgeführt wird, wenn Objekt nicht geladen werden kann

Beispiel:

```
zeiger_auf_applet.altHTML="<B>Applet nicht geladen !<B>";
```

.altKey

ALT-Tasten-Status

Objekt event

false ALT-Taste ist nicht gedrückt

true ALT-Taste ist gedrückt

Beispiel:

```
<HEAD>
<SCRIPT>
function InnerTextSetzen(Zeiger, Kette)
{Zeiger.innerText=Kette;}

function init()
{
    InnerTextSetzen(ID_Span1,'false');
    InnerTextSetzen(ID_Span2,'false');
}
function AltDown()
{
    if (event.altLeft)
    { InnerTextSetzen(ID_Span1,'true'); }
    else
    {
        if (event.altKey)
        { InnerTextSetzen(ID_Span2,'true'); }
    }
}

function AltUp()
{
    if (!event.altKey)
    {
        InnerTextSetzen(ID_Span1,'false');
        InnerTextSetzen(ID_Span2,'false');
    }
}
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="AltDown();" onkeyup="AltUp();">
<TABLE>
<TR>
<TD><I>Linke ALT-Taste gedruickt</I></TD>
<TD><I>Recchte ALT-Taste gedruickt</I></TD>
</TR>
<TR>
<TD ALIGN="center">
<SPAN ID="ID_Span1"></SPAN>
</TD>
<TD ALIGN="center">
<SPAN ID="ID_Span2"></SPAN>
</TD>
</TR>
</TABLE>
</BODY>
```

.altLeft

linke ALT-Taste-Status

Objekt event

nicht für Win9x

nur für Dokument, das den Fokus hat

false linkeALT-Taste ist nicht gedrückt

true linke ALT-Taste ist gedrückt

Beispiel:

```
<HEAD>
<SCRIPT>
function InnerTextSetzen(Zeiger, Kette)
{Zeiger.innerText=Kette;}

function init()
{
```



```

        InnerTextSetzen(ID_Span1,'false');
        InnerTextSetzen(ID_Span2,'false');
    }
    function AltDown()
    {
        if (event.altLeft)
        { InnerTextSetzen(ID_Span1,'true'); }
        else
        {
            if (event.altKey)
            { InnerTextSetzen(ID_Span2,'true'); }

        }
    }

    function AltUp()
    {
        if (!event.altKey)
        {
            InnerTextSetzen(ID_Span1,'false');
            InnerTextSetzen(ID_Span2,'false');
        }
    }
}
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="AltDown();" onkeyup="AltUp();">
    <TABLE>
        <TR>
            <TD><I>Linke ALT-Taste gedrueckt</I></TD>
            <TD><I>Recchte ALT-Taste gedrueckt</I></TD>
        </TR>
        <TR>
            <TD ALIGN="center">
                <SPAN ID="ID_Span1"></SPAN>
            </TD>
            <TD ALIGN="center">
                <SPAN ID="ID_Span2"></SPAN>
            </TD>
        </TR>
    </TABLE>
</BODY>

```

.appName	<p>Codename des Browsers per navigator Objekt</p> <p>navigator.appCodeName</p> <p>"Mozilla" Default</p> <p>geliefert von IE und NS</p> <p>sonst browserspezifisch</p> <p>Achtung Opera-Browser gibt sich als IE aus !</p>
.appCodeName	<p>Codename des Browsers</p> <p>siehe Objekt window.clientInformation</p> <p>"Mozilla" Default</p> <p>geliefert von IE und NS</p> <p>sonst browserspezifisch</p> <p>Achtung Opera-Browser gibt sich als IE aus !</p>
APPLICATION	<p>Umgebung des Frames mit Vertrauen-Status im Rahmen des Browser-Sicherheitsmodells</p> <p>Attribut wird nicht vererbt an Kinder-Frame</p> <p>"no" Default, keine Vertrautheit</p> <p>"yes" Vertrautheit</p>
.appMinorVersion	<p>Update der Browserversion per navigator Objekt</p> <p>z.B. 1 des IE 4.1</p> <p>navigator.appMinorVersion</p>
.appVersion	<p>Browser-Versionsnummer: Unternummer der Hauptnummer</p> <p>z.B. 1 des IE 4.1</p> <p>siehe Objekt window.clientInformation</p>
.appName	<p>Name des Browsers per navigator Objekt</p> <p>navigator.appName</p> <p>"Microsoft Internet Explorer" Default</p> <p>"Netscape" ist der Netscape Navigator</p>

Beispiel:

```
<SCRIPT LANGUAGE="JavaScript">
```




```

<!--      var ie4=0;
           if (      (navigator.appName.indexOf("Explorer")>=0)
           &&      (navigator.appVersion.indexOf('4.0')    >=0)
           )
           ie4=1;
// -->
</SCRIPT>

```

für Netscape bitte "Netscape" verwenden und entsprechende Version

.appName Browsername
 siehe Objekt window.clientInformation
 "Microsoft Internet Explorer" Default
 "Netscape" ist der Netscape Navigator

.appVersion Betriebssystem-Plattform und Browserversion per navigator Objekt
 navigator.appVersion
 Aufbau

 clientVersion (platform; information; extraInformation)
 oder 5.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)
 Hinweis: Haupt-Versionsnummer ermittelbar per parseFloat(navigator.appVersion)
 z.B. 4 des IE 4.1

Beispiel:

```

<SCRIPT LANGUAGE="JavaScript">
<!--      var ie4=0;
           if (      (navigator.appName.indexOf("Explorer")>=0)
           &&      (navigator.appVersion.indexOf('4.0')    >=0)
           )
           ie4=1;
// -->
</SCRIPT>

```

für Netscape bitte "Netscape" verwenden und entsprechende Version

.appVersion Plattform und Browserversion
 Aufbau

 clientVersion (platform; information; extraInformation)
 oder 5.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)
 Hinweis: Haupt-Versionsnummer ermittelbar per parseFloat(navigator.appVersion)
 z.B. 4 des IE 4.1
 siehe Objekt window.clientInformation

.archive Zeichenkette für Archive-Funktionalität eines Objektes

.arguments Zeiger auf das Script-Objekt arguments
 alle Eigenschaften des Script-Objektes arguments können referenziert werden
 arguments nur verfügbar während der Ausführung der Funktion
 siehe Script-Objekt Funktion und Script-Objekt arguments und Anweisung function

Beispiel:

```

function Test1()
{
    var AnzahleArgumente = arguments.length;
    var Kette = "Anzahl Argumente = " + AnzahleArgumente + "\n";

    for ( var i = 0; i < AnzahleArgumente; i++)
    { Kette = "Argument[" + i + "] = " + arguments[i] + "\n"; }

    alert(Kette);
}

function Test2(Wert1,Wert2)
{
    var AnzahleArgumente = arguments.length;
    var Kette = "Anzahl Argumente = " + AnzahleArgumente + "\n";

    for ( var i = 0; i < AnzahleArgumente; i++)
    { Kette = "Argument[" + i + "] = " + arguments[i] + "\n"; }

    alert(Kette);
}

Test1();
Test2(10,20);

```



.AtEndOfLine Satzzeiger bezüglich Zeilenende (End Of Line (EOL)) per newline-Zeichen
wird mit jedem Lesen oder Schreiben verändert
Hinweis: newline-Zeichen per `.WriteLine` oder `.WriteBlankLines()` schreibbar
immer bei satzweisem Schreiben/Lesen
möglich beim zeichenweisem Schreiben/Lesen

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
var Kette = "";
while (!DateiOffen.AtEndOfLine)
{ Kette = Kette + DateiOffen.ReadLine() + "\n"; }
DateiOffen.Close();
alert(Kette);
```

.AtEndOfStream Satzzeiger bezüglich Dateiende, also Textstream-Ende (End Of Stream (EOS))
wird mit jedem Lesen oder Schreiben verändert

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
var Kette = "";
while (!DateiOffen.AtEndOfStream)
{ Kette = Kette + DateiOffen.ReadLine() + "\n"; }
DateiOffen.Close();
alert(Kette);
```

ATOMICSELECTION Selektierbarkeit des Objektes einstellen
false für nicht selektierbar, Default
true für selektierbar, aber Dokumentdarstellung ist langsamer als bei false

.attributeCount Anzahl der mit dem MediaItem Objekt verbundenen Attribute liefern
Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:
`.getItemInfo()`
`.getAttributeName()`
Eigenschaften nutzbar:
`.attributeCount`
Es ist also vorher die Eigenschaft `.playState` auf Werte > 10 zu prüfen.
siehe Behavior `.style.mediaBar`

Beispiel:

```
<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick="ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
ID_Div.enabled = false;
"
>
```

.attributeName Bezeichner einer Style-Eigenschaft, also Name eines Attributes, für die Animation anhand dieses Attributes auf der Timeline
siehe `.accumulate` `.by` `.additive` `.calcMode`
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft `.width`) nach rechts:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function SpanInhaltInit(Kette)
{
// Zustand
ID_Span1.innerText = "";

// Kumulationsart
ID_Span2.innerText =Kette;
```



```

        // Zaehler auf 1
        ID_Span3.innerText ="1";

        // DIV-Text festlegen
        ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerText += " genau 1x kumulativ um ";
            ID_Div.innerText += ID_Animate.by
            ID_Div.innerText += " mal "
            ID_Div.innerText += ID_Animate.repeatCount;
        }
        else
        {
            // keine Wertkumulation von .width
            ID_Div.innerText += ID_Animate.repeatCount;
            ID_Div.innerText += " mal um ";
            ID_Div.innerText += ID_Animate.by
        }

        ID_Div.innerText += " aus";
    }

    function KumulationAus()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("none");
        ID_Animate.accumulate="none";

        // DANACH Animation starten
        ID_Animate.beginElement();

    }

    function KumulationEin()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("sum");
        ID_Animate.accumulate="sum";

        // DANACH Animation starten
        ID_Animate.beginElement();

    }

    function Anzeige()
    {
        ID_Span1.innerText = "Animation ist beendet ";
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE
        ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

    animierter DIV
    <DIV
        ID="ID_Div"
        CLASS="time_line_klasse"
        STYLE= "position:absolute;
                top:125px;

```



```

        left:25px;
        height:100px;
        width:125px;
        border:solid black 2px;
        "
    >
</DIV>
<BR>

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3"></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.Attributes Attribute des Ordner

Beispiel:

```

var OrdnerName                      = "c:\\windows\\desktop\\";
var DateiSystem                    = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                          = DateiSystem.GetFolder(OrdnerName);
var Ordner_Attribute               = Ordner.attributes;

// auf gesetztes Archivattribut prüfen
if (Ordner_Attribute && 32)
{
    // löschen
    Ordner_Attribute -= 32;
}
else
{
    // setzen
    Ordner_Attribute += 32;
}

```

.Attributes Attribute der Datei

Beispiel:

```

var DateinameMitPfad               = "c:\\test.txt";
var DateiSystem                    = new ActiveXObject("Scripting.FileSystemObject");
var Datei                          = DateiSystem.GetFile(DateinameMitPfad);
var Datei_Attribute               = Datei.attributes;

// auf gesetztes Archivattribut prüfen
if (Datei_Attribute && 32)
{
    // löschen
    Datei_Attribute -= 32;
}
else
{
    // setzen
    Datei_Attribute += 32;
}

```

.author

Name des Autor der Media-Datei auf der Timeline

bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von AUTHOR des aktiven Eintrages geliefert und nicht den der Datei.

Track entspricht playItem Objekt laut object.playlist.item() bzw. object.playlist.aktiveTrack
Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```
var Kette = object.playlist.item(index).author;
```



index Integer, ab 0

```
var Kette = object.playList.aktiveTrack.author;
```

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                     SRC="test.wmv"
                     STYLE="position:absolute;top:50px;height:100px"
    >
    </t:VIDEO>
    <SPAN ID="ID_Span"
        STYLE="position:absolute;top:165px;"
    >
    </SPAN>
    <BUTTON ID="ID_Button"
        onclick="ID_Span.innerText= ID_Video.author"
    >
        Klick
    </BUTTON>
</BODY>
</HTML>
```

Beispiel 3:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playList.aktiveTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playList.aktiveTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playList.aktiveTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playList.aktiveTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playList.aktiveTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playList.aktiveTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playList.aktiveTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playList.aktiveTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
    }
</SCRIPT>
</HTML>
```



```

        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA          ID="ID_Media"
                     SRC="test.asx"
                     BEGIN="indefinite"
                     TIMEACTION="visibility"
                     onend="ButtonUpdate();"
                     ontrackchange="AnzeigeUpdate();"
                     onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN  ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN  ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN  ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN  ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN  ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN  ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN  ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN  ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON          ID="ID_Button1"
                     onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON          ID="ID_Button2"
                     onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON          ID="ID_Button3"
                     onclick="ID_Media.playlist.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON          ID="ID_Button4"
                     onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>
</HTML>

```

.autocomplete Status des Autocomplete (Autovervollständigung) zum Formular
 Einstellung der Autovervollständigung im Browser per
 Extras-Internetoptionen-Inhalte
Achtung: Missbrauch bei input password Objekt möglich !
 "on" oder "off"

Beispiel:

```
<INPUT TYPE="password" AUTOCOMPLETE="off">
```

Hinweise: AutoComplete betrifft folgende Werte:

laut VALUE-Attribute von Textfeldern im Formular, aber NUR, wenn diese
 Felder auch das NAME-Attribut besitzen, wobei VALUE-Werte
 automatisch für die automatische Wiederverwendung im Browser
 gespeichert werden (auch bei Password-Feldern !!!)



Werte aus dem vCard-Schema laut dem Objekt navigator.userProfile verwendet
(siehe dort)

vCard.XXX

vCard.Business.City
vCard.Business.Country
vCard.Business.Fax
vCard.Business.Phone
vCard.Business.State
vCard.Business.StreetAddress
vCard.Business.URL
vCard.Business.Zipcode
vCard.Cellular
vCard.Company
vCard.Department
vCard.DisplayName
vCard.Email
vCard.FirstName
vCard.Gender
vCard.Home.City
vCard.Home.Country
vCard.Home.Fax
vCard.Home.Phone
vCard.Home.State
vCard.Home.StreetAddress
vCard.Home.Zipcode
vCard.Homepage
vCard.JobTitle
vCard.LastName
vCard.MiddleName
vCard.Notes
vCard.Office
vCard.Pager

Die Nutzung von AutoComplete sollte reiflich überlegt sein.

Das nachträgliche Löschen von per AutoComplete automatisch gespeicherte Werte
ist in den Internet-Optionen des IE vollziehbar.

.autoReverse automatische Rückwärtsanimation eines Elementes nach dem kompletten Ende der Vorwärtsanimation
des Elementes auf der Timeline
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
function HochZaehlen()
{
    ID_Div1.innerText = "Zähler: "
                        + (ID_Animation.currTimeState.repeatCount + 1);
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div1"></DIV>
<DIV ID="ID_Div2"
CLASS="time_line_klasse"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<t:ANIMATEMOTION
ID="ID_Animation"
TARGETELEMENT="ID_Div2"
TO="150,0"
DUR="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
onrepeat="HochZaehlen()"
>
</t:ANIMATEMOTION>
</BODY>
</HTML>
```



.AvailableSpace Freien Speicher in Bytes auf Laufwerk ermitteln

Beispiel:

```
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk         = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_FreierPlatz = Laufwerk.AvailableSpace;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_FreierPlatz);
```

.availHeight verfügbare Höhe des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar
per screen Objekt

.availHeight verfügbare Höhe des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar
Behavior .style.clientCaps

.availWidth verfügbare Breite des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar
per screen Objekt

.availWidth verfügbare Breite des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar
Behavior .style.clientCaps

.background Hintergrundbild im Dokument
document.body.background

.background Hintergrundbild eines Objektes

.balance Balance im Objekt BGSOUND
-10 bis +10
-10 ganz links
+10 ganz rechts
0 genau mittig, default

Beispiel:

```
<HEAD>
<SCRIPT>
function KanalVerteilung(Wert)
{ ID_bgsound.balance = Wert; }

function GenauEinmal()
{
    ID_bgsound.loop = 1;
    ID_bgsound.src = ID_bgsound.src; // restart
}

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
<BUTTON onclick="GenauEinmal()"></BUTTON>
<BUTTON onclick="Endlos()"></BUTTON>
<BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(0)"></BUTTON>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-10;"
>Mute

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-7;"
>25% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_bgsound.volume=-5;"
>50% Volume

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_bgsound.volume=-2;"
>75% Volume
```



```

        < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
            onpropertychange="ID_bgsound.volume=0;"
            >100% Volume
    </BODY>

```

.Banner

Text laut BANNER des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playList
 ab IE 6.x
 Track entspricht playItem Objekt
 Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```
var Kette = object.playList.aktiveTrack.Banner;
```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playList.aktiveTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playList.aktiveTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playList.aktiveTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playList.aktiveTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playList.aktiveTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playList.aktiveTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playList.aktiveTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playList.aktiveTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"

```



```

TIMEACTION="visibility"
onend="ButtonUpdate();"
ontrackchange="AnzeigeUpdate();"
onmediacomplete="ButtonUpdate();AnzeigeUpdate();"

>
</t:MEDIA>

<SPAN ID="ID_Span1">Titel:</SPAN>
<BR>
<SPAN ID="ID_Span2">Autor:</SPAN>
<BR>
<SPAN ID="ID_Span3">Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span4">Copyright:</SPAN>
<BR>
<SPAN ID="ID_Span5">Filename:</SPAN>
<BR>
<SPAN ID="ID_Span6">Banner:</SPAN>
<BR>
<SPAN ID="ID_Span7">Banner Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

<BR>
<BUTTON ID="ID_Button1"
onclick="ID_Media.beginElement(); ButtonUpdate();"

>
Start
</BUTTON>
<BUTTON ID="ID_Button2"
onclick="ID_Media.playList.nextTrack();"

>
naechster Track
</BUTTON>
<BUTTON ID="ID_Button3"
onclick="ID_Media.playList.prevTrack();"

>
vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
onclick="ID_Media.endElement(); AnzeigeLoeschen();"

>
Stop
</BUTTON>
</BODY>
</HTML>

```

.BannerAbstract Text laut BANNER und dort ABSTRACT des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection `.style.time2.playList` ab IE 6.x
Track entspricht `playItem` Objekt
Trackliste liegt in der Behavior-Collection `.style.time2.playList` als Playliste:
Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```
var Kette = object.playList.aktiveTrack.BannerAbstract;
```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
function ButtonUpdate()
{
    if (ID_Media.currTimeState.isActive)
    {
        ID_Button1.disabled=true;
        ID_Button2.disabled=false;
        ID_Button3.disabled=false;
    }
}

```



```

        ID_Button4.disabled=false;
    }
    else
    {
        ID_Button1.disabled=false;
        ID_Button2.disabled=true;
        ID_Button3.disabled=true;
        ID_Button4.disabled=true;
    }
}

function AnzeigeUpdate()
{
    ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
    ID_Span3.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
    ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
    ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
    ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
    ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
    ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
    ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
}

function AnzeigeLoeschen()
{
    ID_Span1.innerText = "Titel: ";
    ID_Span2.innerText = "Autor: ";
    ID_Span3.innerText = "Abstract: ";
    ID_Span4.innerText = "Copyright: ";
    ID_Span5.innerText = "Filename: ";
    ID_Span6.innerText = "Banner: ";
    ID_Span7.innerText = "Banner Abstract: ";
    ID_Span8.innerText = "Banner MoreInfo: ";
}
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"
        TIMEACTION="visibility"
        onend="ButtonUpdate();"
        ontrackchange="AnzeigeUpdate();"
        onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON ID="ID_Button2"
        onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track

```



```

</BUTTON>
<BUTTON ID="ID_Button3"
        onclick="ID_Media.playlist.prevTrack();"
>
        vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"
>
        Stop
</BUTTON>
</BODY>
</HTML>

```

.BannerMoreInfo Text laut BANNER und dort MOREINFO des aktiven Eintrages in einer Advanced Stream Redirector (ASX) –Datei anhand Behavior-Collection .style.time2.playlist
 ab IE 6.x
 Track entspricht playItem Objekt
 Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:
 var Kette = object.playlist.aktiveTrack.BannerMoreInfo;

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
function ButtonUpdate()
{
    if (ID_Media.currTimeState.isActive)
    {
        ID_Button1.disabled=true;
        ID_Button2.disabled=false;
        ID_Button3.disabled=false;
        ID_Button4.disabled=false;
    }
    else
    {
        ID_Button1.disabled=false;
        ID_Button2.disabled=true;
        ID_Button3.disabled=true;
        ID_Button4.disabled=true;
    }
}

function AnzeigeUpdate()
{
    ID_Span1.innerText = "Titel: " + ID_Media.playlist.aktiveTrack.title;
    ID_Span3.innerText = "Autor: " + ID_Media.playlist.aktiveTrack.author;
    ID_Span3.innerText = "Abstract: " + ID_Media.playlist.aktiveTrack.abstract;
    ID_Span4.innerText = "Copyright: " + ID_Media.playlist.aktiveTrack.copyright;
    ID_Span5.innerText = "Filename: " + ID_Media.playlist.aktiveTrack.src;
    ID_Span6.innerText = "Banner: " + ID_Media.playlist.aktiveTrack.Banner;
    ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.aktiveTrack.BannerAbstract;
    ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.aktiveTrack.BannerMoreInfo;
}

function AnzeigeLoeschen()
{
    ID_Span1.innerText = "Titel: ";
    ID_Span2.innerText = "Autor: ";
    ID_Span3.innerText = "Abstract: ";
    ID_Span4.innerText = "Copyright: ";
    ID_Span5.innerText = "Filename: ";
    ID_Span6.innerText = "Banner: ";
    ID_Span7.innerText = "Banner Abstract: ";
    ID_Span8.innerText = "Banner MoreInfo: ";
}

```



```

    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA          ID="ID_Media"
                     SRC="test.asx"
                     BEGIN="indefinite"
                     TIMEACTION="visibility"
                     onend="ButtonUpdate();"
                     ontrackchange="AnzeigeUpdate();"
                     onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN   ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN   ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN   ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN   ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN   ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN   ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN   ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN   ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON          ID="ID_Button1"
                     onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON          ID="ID_Button2"
                     onclick="ID_Media.playList.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON          ID="ID_Button3"
                     onclick="ID_Media.playList.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON          ID="ID_Button4"
                     onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>
</HTML>

```

.BaseHref Url des Dokumentes, das <OBJECT> </OBJECT> enthält (auch frame und iframe)
ist meistens die Url des Dokumentes selbst

.begin Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction
siehe Objekt currTimeState und Behavior .style.time2
Wert im Time-Format
z.B. "h:min:s.f"
 id.event
 id.event+zeit_wert_als_string

Beispiele für Kodierung von Time in der Eigenschaft .begin:

```

object.begin="objekt_zeiger.begin+10s"           // warten bis Event "onbegin" zum Objekt laut
                                                    //          objekt_zeiger (z.B. laut ID-Attribut)
                                                    //          eintritt,
                                                    //          dann 10 Sekunden warten
                                                    //          dann Objekt laut object starten
object.begin="objekt_zeiger.focus+10s"           // warten bis Event "onfocus" zum Objekt laut
                                                    //          objekt_zeiger (z.B. laut ID-Attribut)

```



```

//      eintritt,
//      dann 10 Sekunden warten
//      dann Objekt laut object starten

object.begin="2; objekt_zeiger.click+1" // 2 Sekunden nach dem Laden des Elternobjektes von
//      object warten
//      dann auf das Ereignis click zum Objekt laut
//      objekt_zeiger warten
//      dann 1 Sekunde warten
//      dann Objekt laut object starten

```

Hinweis: object laut ID-Attribut des Behavior-Objektes von .style.time2.

```

"25:45:10" 25 Stunden, 45 Minuten, 10 Sekunden
"45:35"    45 Minuten, 35 Sekunden
"45:00.275" 45 Minuten, 0,275 Sekunde
"10.5"     10,5 Sekunden

```

Beispiele für Kodierung von .begin:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY BGCOLOR="white">
  <SPAN ID="ID_Span1"
        CLASS=time_line_klasse
        STYLE="COLOR:Red;"
        BEGIN="0" DUR="3" TIMEACTION="display"
  >
    <H3>Test2</H3>
  </SPAN>
  <SPAN ID="ID_Span2"
        CLASS=time_line_klasse
        STYLE="COLOR:Blue;" BEGIN="4" DUR="4"
        TIMEACTION="display"
  >
    <H3>Test2</H3>
  </SPAN>
  <H1 ID="ID_H1"
      CLASS=time_line_klasse
      BEGIN="8"
      DUR="indefinite"
      TIMEACTION="display"
  >
    Ende
  </H1>
</BODY>
</HTML>

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
  <P>Klick Button: 2 Sekunden nach Klick wird Span fuer 5 Sekunden angezeigt</P>
  <BR>
  <BUTTON ID="ID_Button">Klick mich</BUTTON>
  <BR>
  <SPAN ID="ID_Span"
        CLASS=time_line_klasse
        BEGIN="ID_Button.click+2"
        DUR="5"
        TIMEACTION="display"
        STYLE="COLOR:Red;"
  >

```




```

        </SPAN>
        </BODY>
    </HTML>

```

.behavior Verhalten des Scrollens des marquee Objektes

"scroll" Default
 Scrollrichtung laut Eigenschaft .direction
 wenn Grenze des Elterncontainer erreicht wurde, wird an der entgegengesetzten Grenze wieder in den Elterncontainer reingescrollt

"alternate" Scrollrichtung umkehren sobald Grenze des Elterncontainer erreicht wurde

"slide" Scrollrichtung laut Eigenschaft .direction
 wenn Grenze des Elterncontainer erreicht wurde, wird rausgescrollt und dann Scrollen gestoppt

Beispiel:

```

<MARQUEE LOOP=1
    HEIGHT=200
    WIDTH=740
    SCROLLAMOUNT=10
    SCROLLDELAY=20
    BEHAVIOR="SLIDE"
    DIRECTION="DOWN"
    STYLE="position:absolute; top:0; left:10"
>
    Dieser Text scrollt
</MARQUEE>

```

.bgColor deprecated und durch STYLE-Attribut zu ersetzen
 Hintergrundfarbe des Objektes bzw. Dokumentes (document.body)
 #rrggbb Standard ist #0000FF
 vordefinierter Farbname (browserspezifisch)

.bgProperties Scroll-Eigenschaften des Hintergrundbildes im Dokument
 "" für scrollen
 "fixed" für nichtscrollen

Beispiel:

```

<BODY BACKGROUND="/images/test.gif" BGPROPERTIES="fixed">

```

.blockDirection Umfluss um ein Objekt
 "ltr" Umfluss von links nach rechts
 "rtl" Umfluss von rechts nach links

.border Rahmendicke in Pixel

.borderColor Borderfarbe (Rahmenfarbe)
 wird ignoriert wenn Eigenschaft .frameBorder mit Wert 0 oder "no"
 Eigenschaft .border mit Wert 0
 #rrggbb
 vordefinierter Farbname (browserspezifisch)

.borderColorDark deprecated und durch Eigenschaft .borderColor zu ersetzen
 dunkle Farbe des 3D-Rahmens eines Objektes
 zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)

.borderColorLight deprecated und durch Eigenschaft .borderColor zu ersetzen
 helle Farbe des 3D-Rahmens eines Objektes
 zu verwenden mit Eigenschaft .border (nicht Style-Eigenschaft border)

.bottom untere Pixelposition des Rechteckes um ein Objekt
 auch textrectangle Objekt

Beispiel für Nicht-TextRectangle-Objekt:

```

<SCRIPT>
    function Anzeigen(ObjektZeiger)
    {
        var Rechteck = ObjektZeiger.getBoundingClientRect();

        alert(
            "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
            + "\n"
            + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
        );
    }
</SCRIPT>
<P onclick="Anzeigen(this)">

```



Beispiel für TextRectangleObjekt:

```
<SCRIPT>
function Anzeigen(ObjektZeiger)
{
    var TextRectangleCollection = document.body.ObjektZeiger.TextRange.getClientRechts();

    // .getClientRechts() liefert immer Collection der textrectangle Objekte !!!

    var Rechteck = TextRectangleCollection[0];

    alert(    "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
            + "\n"
            + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
            );
}
</SCRIPT>
<BODY>
<DIV onclick="Anzeigen(this)">
    Test
</DIV>
</BODY>
```

.bottomMargin Bottom Margin des Dokumentes in Pixel
 Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

BOUNDARY wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software

.boundingHeight Höhe des Rechteckes in Pixel um den Textbereich des TextRange Objektes
 per textrange Objekt

Beispiel:

```
<SCRIPT>
function Anzeige(ZeigerAufTextarea)
{
    var FeldAllterTextArea = document.all.tags("TEXTAREA");

    // prüfen ob mindestens 1 TEXTAREA-Element im Body vorliegt
    if (FeldAllterTextArea != null)
    {
        var TextBereich = ZeigerAufTextarea.createTextRange();

        // prüfen ob Textbereich erzeugt wurde
        if (TextBereich != null)
        { alert(".boundingHeight = " + TextBereich.boundingHeight); }
    }
}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS=100 ROWS=2 onclick="Anzeige(this)">
    .....
</TEXTAREA>
```

.boundingLeft Abstand linker Rand des Rechteckes in Pixel um den Textbereich zur linkem Rand des Container-Objektes,
 in dem der Textbereich liegt
 per textrange Objekt

Beispiel:

```
<SCRIPT>
function Anzeige(ZeigerAufTextarea)
{
    var FeldAllterTextArea = document.all.tags("TEXTAREA");

    // prüfen ob mindestens 1 TEXTAREA-Element im Body vorliegt
    if (FeldAllterTextArea != null)
    {
        var TextBereich = ZeigerAufTextarea.createTextRange();

        // prüfen ob Textbereich erzeugt wurde
        if (TextBereich != null)
        { alert(".boundingLeft = " + TextBereich.boundingLeft); }
    }
}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS=100 ROWS=2 onclick="Anzeige(this)">
```



```
.....
</TEXTAREA>
```

.boundingTop Abstand oberer Rand des Rechteckes in Pixel um den Textbereich zum oberen Rand des Container-Objektes, in dem der Textbereich liegt per textrange Objekt

Beispiel:

```
<SCRIPT>
function Anzeige(ZeigerAufTextarea)
{
    var FeldAllerTextArea = document.all.tags("TEXTAREA");

    // prüfen ob mindestens 1 TEXTAREA-Element im Body vorliegt
    if (FeldAllerTextArea != null)
    {
        var TextBereich = ZeigerAufTextarea.createTextRange();

        // prüfen ob Textbereich erzeugt wurde
        if (TextBereich != null)
        {alert(".boundingTop = " + TextBereich.boundingTop); }
    }
}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS=100 ROWS=2 onclick="Anzeige(this)">
.....
</TEXTAREA>
```

.boundingWidth Breite des Rechteckes in Pixel um den Textbereich des TextRange Objektes per textrange Objekt

Beispiel:

```
<SCRIPT>
function Anzeige(ZeigerAufTextarea)
{
    var FeldAllerTextArea = document.all.tags("TEXTAREA");

    // prüfen ob mindestens 1 TEXTAREA-Element im Body vorliegt
    if (FeldAllerTextArea != null)
    {
        var TextBereich = ZeigerAufTextarea.createTextRange();

        // prüfen ob Textbereich erzeugt wurde
        if (TextBereich != null)
        {alert(".boundingWidth = " + TextBereich.boundingWidth); }
    }
}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS=100 ROWS=2 onclick="Anzeige(this)">
.....
</TEXTAREA>
```

.browserLanguage Sprache des Betriebssystems und **nicht** des Browsers (Browsersprache kann andere sein als die des Betriebssystems z.B. französischer Browser auf deutschem Windows) per navigator Objekt
Achtung: ab IE 5.x Sprache **immer** laut **regionalen** Einstellungen des Users im **Betriebssystems** z.B. "en" oder "de"

.browserLanguage Browsersprache
Achtung: ab IE 5.x Sprache **immer** laut **regionalen** Einstellungen des Users im **Betriebssystems** z.B. "en" oder "de"
siehe Objekt window.clientInformation

.bufferDepth Anzahl der Bits pro Pixel für eine Farbe UND Verwendung des off-screen bitmap buffer per screen Objekt

.bufferDepth Anzahl der Bits pro Pixel für eine Farbe UND Verwendung des off-screen bitmap buffer Behavior .style.clientCaps

.bufferingProgress aktueller prozentualer Status des Pufferns eines Datenflusses (Data Stream) einer Media-Datei auf der Timeline
Prozentwert des bisher erfolgten Pufferns
nur für Media-Datei mit Datenfluss
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
```



```

<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:MEDIA          ID="ID_Media"
                  SRC="test.asx"
                  SYNCBEHAVIOR="locked"
                  TIMEACTION="display"
>
</t:MEDIA>
<SPAN  ID="ID_Span"
      CLASS="time_line_klasse"
      DUR="0.1"
      REPEATCOUNT="indefinite"
      onrepeat="ID_Span.innerText= ID_Media.bufferingProgress;"
>
0
</SPAN>
</BODY>
</HTML>

```

.button Maustaste die das Event auslöst NUR per onmousedown, onmouseup, und onmousemove events
Objekt event
Integer

0	Default. keine Taste gedrückt oder obiges Event nicht erzeugt
1	linke Maustaste ist gedrückt
2	rechte Maustaste ist gedrückt
3	linke UND rechte Maustaste sind gleichzeitig gedrückt
4	mittlere Maustaste ist gedrückt
5	linke UND mittlere Maustaste sind gleichzeitig gedrückt
6	rechte UND mittlere Maustaste sind gleichzeitig gedrückt
7	rechte UND mittlere UND linke Maustaste sind gleichzeitig gedrückt

.by Wert um den die Werterhöhung bei Elemente-Animation(en) auf der Timeline
per .additive oder .accumulate
erfolgen soll
Wert der Eigenschaft:
numerisch aber Einheiten wie px, in, cm, mm, pt, pc können Teil des Wertes sein
wird für Animation in Einheiten zerlegt laut Schrittweite laut .calcMode
für die Objekte animate, animateMotion und animateColor gilt:
.by wird von .path, .to und .values überschrieben
Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
siehe .attributeName .accumulate .additive .calcMode
siehe Objekt currTimeState und Behavior .style.time2

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function SpanInhaltInit(Kette)
{
    // Zustand
    ID_Span1.innerText = "";

    // Kumulationsart
    ID_Span2.innerText =Kette;

    // Zaehler auf 1
    ID_Span3.innerText = "1";

    // DIV-Text festlegen
    ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
    if (Kette == "sum")
    {
        // Wertkumulation von .width
        ID_Div.innerText += " genau 1x kumulativ um ";
    }
}

```



```

        ID_Div.innerText += ID_Animate.by
        ID_Div.innerText += " mal "
        ID_Div.innerText += ID_Animate.repeatCount;
    }
    else
    {
        // keine Wertkumulation von .width
        ID_Div.innerText += ID_Animate.repeatCount;
        ID_Div.innerText += " mal um ";
        ID_Div.innerText += ID_Animate.by
    }

    ID_Div.innerText += " aus";
}

function KumulationAus()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("none");
    ID_Animate.accumulate="none";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function KumulationEin()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("sum");
    ID_Animate.accumulate="sum";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function Anzeige()
{
    ID_Span1.innerText = "Animation ist beendet ";
}
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE
        ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

animierter DIV
<DIV
    ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE= "position:absolute;
            top:125px;
            left:25px;
            height:100px;
            width:125px;
            border:solid black 2px;
            "
    >
</DIV>
<BR>

```



```

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3" ></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.by

Schrittweite des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt
als Offset zum Wert der kodierten Eigenschaft .from des Behavior
Schrittweite als Prozentsatz des Überganges nach dessen Vollendung
nicht zusammen mit Eigenschaft .to oder .value kodieren, sonst wird .by ignoriert
siehe Objekt currTimeState und Behavior .style.time2
Ziffernfolge Floating point
Wert numerisch von 0 bis 1
0 entspricht 0 % des kompletten Überganges
1 entspricht 100% des kompletten Überganges
Bsp.: 0.3 entspricht Schrittweite ist 30 % des kompletten Überganges

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter"
FROM="0.3"
BY="0.4"
TYPE="barWipe"
DUR="3"
TARGETELEMENT="ID_Div"
>
</t:TRANSITIONFILTER>

<DIV ID="ID_Div"
CLASS="time_line_Klasse"
DUR="indefinite"
STYLE="position:relative; left:20px; width:420px; height:100px;
background-image:url(test.gif); background-repeat: no-repeat;
"
>
</DIV>
</BODY>
</HTML>

```

.calcMode

Schrittweite der Werterhöhung bzw. Art der Animation in der Ebene (1D oder 2D)
bei Elemente-Animation(en) auf der Timeline
per .additive oder .accumulate
Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
siehe Objekt currTimeState und Behavior .style.time2

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function SpanInhaltInit(Kette)
{
// Zustand

```



```

        ID_Span1.innerText = "";

        // Kumulationsart
        ID_Span2.innerText =Kette;

        // Zaehler auf 1
        ID_Span3.innerText ="1";

        // DIV-Text festlegen
        ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerText += " genau 1x kumulativ um ";
            ID_Div.innerText += ID_Animate.by
            ID_Div.innerText += " mal "
            ID_Div.innerText += ID_Animate.repeatCount;
        }
        else
        {
            // keine Wertkumulation von .width
            ID_Div.innerText += ID_Animate.repeatCount;
            ID_Div.innerText += " mal um ";
            ID_Div.innerText += ID_Animate.by
        }

        ID_Div.innerText += " aus";
    }

function KumulationAus()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("none");
    ID_Animate.accumulate="none";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function KumulationEin()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("sum");
    ID_Animate.accumulate="sum";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function Anzeige()
{
    ID_Span1.innerText = "Animation ist beendet ";
}
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE
        ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        CALCMODE="linear"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

```




```

animierter DIV
<DIV ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE= "position:absolute;
            top:125px;
            left:25px;
            height:100px;
            width:125px;
            border:solid black 2px;
            "
>
</DIV>
<BR>

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3" ></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.calcMode

Interpolation der Aufteilung der internen Animationsschritte auf

Schrittweite	laut .by
Schritt	laut.from bei optionalen .to
Gesamtschritt	bei fehlendem .by bzw. .from .to
Animationsschritte	laut .values

per Übergangfilter per .style.time2.transitionFilter Behavior-Objekt

Bsp.: Lineare Aufteilung durch Interpolation: Interne Schritte zu gleichen Teilen verteilt

keine Interpolation, so Animation laut Schrittweite, also eventuell nicht fließende Animation

Größe des internen Animationsschrittes: Je nach der Gesamtdauer der Animation laut .dur des Behavior

siehe Objekt currTimeState und Behavior .style.time2

"discrete" keine Interpolation

"linear" Default

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

<t:TRANSITIONFILTER ID="ID_Transfilter1"
    BEGIN="ID_Div1.begin"
    TYPE="barWipe"
    DUR="5"
    TARGETELEMENT="ID_Div1"
    VALUES=".1;.17;.27;.37;.47;.56;.65;.71;.82;.92;1.0"
    CALCMODE="discrete"
>
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter2"
    BEGIN="ID_Div1.begin"
    TYPE="barWipe"
    DUR="5"
    TARGETELEMENT="ID_Div2"
    VALUES=".1;.17;.27;.37;.47;.56;.65;.71;.82;.92;1.0"
    CALCMODE="linear"
>
</t:TRANSITIONFILTER>
<BR>

```



```
<INPUT TYPE="button" ID="ID_Button" VALUE="Start Transition">

<DIV      ID="ID_Div1"
          CLASS="time_line_Klasse"
          BEGIN="ID_Button.click"
          DUR="indefinite"
          STYLE="position:relative; left:20px; width:420px; height:100px;
                background-image:url(test.gif); background-repeat: no-repeat;
                "
>
</DIV>

<DIV      ID="ID_Div2"
          CLASS="time_line_Klasse"
          BEGIN="ID_Button.click"
          DUR="indefinite"
          STYLE="position:relative; left:20px; width:420px; height:100px;
                background-image:url(test.gif); background-repeat: no-repeat;
                "
>
</DIV>
</BODY>
</HTML>
```

.callee Zeiger auf den Funktionsrumpf
 z.B. verwenden, wenn
 Funktion ohne Funktionsbezeichner erzeugt wurde
 für Kodierung einer Rekursion ohne den Funktionsbezeichner aber **mit Argumentenliste**
 (falls Argumentenliste vorhanden ist)
 siehe Script-Objekt Funktion und Script-Objekt arguments und Anweisung function

```
Beispiel:
function Test(n)
{
    if (n <= 0)
    {return 1;}
    else
    {return (n * arguments.callee(n - 1));}    // Rekursion per arguments.callee(n - 1)
}

alert(Test(3));
```

.caller	<p>Zeiger auf den Aufrufer der Funktion</p> <p>Aufrufer ist eine andere Funktion (außer bei Rekursion)</p> <p>siehe Script-Objekt Funktion und Script-Objekt arguments und Anweisung function</p>
---------	---

<code>.cancelBubble</code>	Event durchreichen über Eventhandlerkette Objekt event false Default. Bubbling ein, also Event in der Eventhandler-Hierarchie durchreichen true kein Durchreichen
----------------------------	--

```
Beispiel:
<SCRIPT LANGUAGE="JScript">
    function DurchReichen()
    {
        if (window.event.shiftKey)
        {window.event.cancelBubble = true;}
    }

    function Anzeigen()
    {
        if (window.event.srcElement.tagName == "IMG")
        {alert(window.event.srcElement.src);}
    }
</SCRIPT>
<BODY onclick="Anzeigen()">
<IMG SRC="test.gif" onclick="DurchReichen()">
```

.canHaveChildren prüfen ob Kind möglich ist, also ob Objekt Parent sein kann

Beispiel:

```
<SCRIPT>
function KindHinzufuegen()
{
    var ZeigerAufSPANObjekt =document.createElement("SPAN");
    var ZeigerAufTextObjekt =document.createTextNode("Test");
```



```

        ZeigerAufSPANObjekt.appendChild(ZeigerAufTextObjekt);

        for(var Index=0; Index <document.all.length; Index ++)
        {
            if(document.all[Index].canHaveChildren==true)
            {
                document.all[Index].appendChild(ZeigerAufSPANObjekt);
                break;
            }
        }
    }
</SCRIPT>
<INPUT TYPE=button VALUE="Kind hinzufügen " onclick="KindHinzufuegen()">
<DIV>
    Test<BR>
</DIV>

```

.canHaveHTML prüfen ob Objekt HTML-Tags enthalten darf

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Antwort(BooleanWert)
    { BooleanWert ? alert("Ja") : alert("Nein");}
</SCRIPT>
</HEAD>
<BODY>
    <P>
        INPUT:
        <INPUT type="text" ID="ID_Input" VALUE="Test" >
    </P>

    <P>
        SPAN:
        <SPAN ID="ID_Span">Test</SPAN>
    </P>

    <BUTTON onclick="Antwort(ID_Input.canHaveHTML);">
        Kann INPUT-Element HTML besitzen ?
    </BUTTON>

    &nbsp;

    <BUTTON onclick="Antwort(ID_Span.canHaveHTML);">
        Kann SPAN-Element HTML besitzen ?
    </BUTTON>
</BODY>
</HTML>

```

.canPause generelle Pausierungsmöglichkeit einer Media-Datei auf der Timeline
siehe Objekt `currTimeState` und Behavior `.style.time2`

.canSeek generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei auf der Timeline
Auswahl per Seek-Methoden
`.seekActiveTime()`
`.seekSegmentTime()`
`.seekTo()`
`.seekToFrame()`
 nicht jeder Media-Typ unterstützt Seek
 siehe Objekt `currTimeState` und Behavior `.style.time2`

.caption Zeiger auf das Objekt `table.caption`
es darf nur 1 CAPTION zur Tabelle existieren

.cellIndex Index der Zelle in der Collection `table.rows.cells`
siehe Objekt `table.tr.td`
siehe Objekt `table.tr.th`

.cellPadding Abstand zwischen Zellrahmen und dem Inhalt der Zelle
siehe Objekt `table`

Beispiel:

```

<TABLE ID="ID_Tabelle" BORDER CELLPADDING=10>
<TR>

```



```

        <TD>Zelle 1</TD>
        <TD>Zelle 2</TD>
    </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.cellPadding=20">20 </BUTTON>
<BUTTON onclick="ID_Tabelle.cellPadding=5">5 </BUTTON>

```

.cellSpacing Abstand zwischen den Zellen
siehe Objekt table

Beispiel:

```

<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
    <TR>
        <TD>Zelle 1</TD>
        <TD>Zelle 2</TD>
    </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.cellSpacing=20">20 </BUTTON>
<BUTTON onclick="ID_Tabelle.cellSpacing=5">5 </BUTTON>

```

.charset Zeichensatz zum Encoden eines Objektes im Dokument

.checked Selektionsstatus des Checkbox-Control bzw. Radio Button-Control
bezüglich Selektion durch User
(Objekt input checkbox bzw. Objekt input radio)
wird bei Ceckbox-Control auch verändert durch Eigenschaft .indeterminate
false Default
keine Selektion des Control-Elementes
true Selektion des Control-Elementes
Hinweise: Der Wert kann nur gesendet werden, wenn das Control selektiert ist.
Werte von unselektierten Controls werden nicht gesendet.
Radio Button-Control nur selektierbar, wenn NAME-Attribut kodiert wurde !

Beispiel:

```

<HEAD>
<SCRIPT>
    function Anzeige()
    {alert("Checkbox-Status = " + ID_Checkbox.checked);}
</SCRIPT>
</HEAD>
<BODY>
    selektiere !
    <INPUT    TYPE=checkbox
        ID="ID_Checkbox"
        NAME="checkbox1"
        onclick=Anzeige()
    >
</BODY>

```

.classid Klassenbezeichner (ID) eines damit dem Objekt zugeordneten ActiveX-Controls von
Microsoft Windows (Microsoft eigene oder Fremdhersteller)
dient als Ersatz für die browserspezifischen MIME-Typen
Gross-Klein egal

Beispiel:

```
"CLSID:6BF52A52-394A-11d3-B153-00C04F79FAA6"
```

.className Klassenreferenz, Klassenname

Beispiel 1:

```

<HEAD>
    <STYLE TYPE="text/css">
        P {font-size: 24pt;}
        .RoterText {color: red;}
        .BlauerText {color: blue;}
        .KursiverText {font-style: italic;}
    </STYLE>
</HEAD>
<BODY>
    <P>fontsize = 24</P>
    <P CLASS="RoterText"> rot UND fontsize = 24</P>
    <P CLASS="BlauerText KursiverText ">blau kursiv UND fontsize = 24</P>
</BODY>

```

Beispiel 2 für globalen Style per HEAD:

```

<HEAD>
<STYLE>

```



```

        .an {text-decoration: underline overline; color:blue;}
        .aus {text-decoration: none; color:black;}
    </STYLE>
</HEAD>
<BODY>
    <A      HREF="test.htm"
           CLASS="aus"
           onmouseover="this.className='an';"
           onmouseout="this.className='aus';"
    >
    </A>
</BODY>

```

.clientHeight Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt
 ohne Rahmen
 ohne Scrollbalken
 Bezüglich des Internet Explorer 6.0 unter Windows 98 und Windows XP bei maximiertem Browserfenster mit Standard-Browser-Menü und einer Bildschirmauflösung von 1024x 768 liefert .clientHeight **unterschiedliche** Werte: Die Höhe unter Windows XP beträgt **10 Pixel mehr** !

Beispiel:

```

<DIV ID="ID_Div" STYLE="overflow:scroll; width:200; height:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientHeight)">client height</BUTTON>

```

.clientInformation Zeiger auf das Objekt window.clientInformation, welches vom Objekt navigator erbt
 siehe Objekt window

.clientLeft Abstand in Pixel zum linken Rand des Fensters

Beispiel:

```

<DIV ID="ID_Div" STYLE="overflow:scroll; top:200; left:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientLeft)">client left</BUTTON>

```

.clientTop Abstand in Pixel zum oberen Rand des Fensters

Beispiel:

```

<DIV ID="ID_Div" STYLE="overflow:scroll; top:200; left:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientTop)">client top</BUTTON>

```

.clientWidth Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt
 ohne Rahmen
 ohne Scrollbalken

Beispiel:

```

<DIV ID="ID_Div" STYLE="overflow:scroll; width:200; height:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientWidth)">client width</BUTTON>

```

.clientX X-Koordinate Maus relativ zum Fenster aber ohne Rahmen, Scrollbar etc.
 Objekt event

Beispiel:

```

<HEAD>
<SCRIPT>
    function AlertAnzeige()
    {
        var Kette = "X= " + window.event.clientX + "\r";
        Kette += "Y= " + window.event.clientY + "\r";
        alert(Kette);
    }

    function StatusZeileAnzeigen()
    {
        var Kette = "X= " + window.event.clientX + " ";
        Kette += "Y= " + window.event.clientY;
        window.status = Kette;
    }
</SCRIPT>
</HEAD>
<BODY onmousemove="StatusZeileAnzeigen()" ondblclick="AlertAnzeige()">
</BODY>

```

.clientY Y-Koordinate Maus relativ zum Fenster aber ohne Rahmen, Scrollbar etc.
 Objekt event

Beispiel:

```

<HEAD>
<SCRIPT>
    function AlertAnzeige()
    {
        var Kette = "X= " + window.event.clientX + "\r";

```



```

        Kette += "Y= " + window.event.clientY + "\r";
        alert(Kette);
    }

    function StatusZeileAnzeigen()
    {
        var Kette = "X= " + window.event.clientX + " ";
        Kette += "Y= " + window.event.clientY;
        window.status = Kette;
    }
</SCRIPT>
</HEAD>
<BODY onmousemove="StatusZeileAnzeigen()" ondblclick="AlertAnzeige()">
</BODY>

```

.clipBegin Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline
Eigenschaft `.canSeek` muss true liefern
nicht jeder Media-Typ unterstützt Clipping
siehe Objekt `currTimeState` und Behavior `.style.time2`

.clipboardData Zeiger auf Objekt `clipboardData` als Windows-Zwischenablage
zu verwenden mit Eventhandlern `onbeforecut` und `onbeforepaste`
siehe auch `event.dataTransfer` Objekt
siehe Objekt `window`

.clipBottom untere Koordinate des Ausschnittes (Clipping Region) per `currentStyle` Objekt
prüfen ob ein Ausschnitt vorliegt

Beispiel:

```

<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect(0,100, "
                        + ID_select.options(ID_select.selectedIndex).value
                        + ",100)";

    if (ID_Image.currentStyle.clipBottom == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>

```

.clipEnd Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes auf der Timeline
Eigenschaft `.canSeek` muss true liefern
nicht jeder Media-Typ unterstützt Clipping
siehe Objekt `currTimeState` und Behavior `.style.time2`

.clipLeft linke Koordinate des Ausschnittes (Clipping Region) per `currentStyle` Objekt
prüfen ob ein Ausschnitt vorliegt

Beispiel:

```

<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect(0,100,100,"
                        + ID_select.options(ID_select.selectedIndex).value
                        + ")";

    if (ID_Image.currentStyle.clipLeft == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>

```



.clipRight rechte Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt

Beispiel:

```
<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect(0,"
                        + ID_select.options(ID_select.selectedIndex).value
                        + ",100,0)";

    if (ID_Image.currentStyle.clipRight == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>
```

.clipTop obere Koordinate des Ausschnittes (Clipping Region) per currentStyle Objekt prüfen ob ein Ausschnitt vorliegt

Beispiel:

```
<SCRIPT>
function AusschnittSetzen()
{
    ID_Image.style.clip= "rect("
                        + ID_select.options(ID_select.selectedIndex).value
                        + ",100,100,0)";

    if (ID_Image.currentStyle.clipTop == "60px")
    { alert("60 Pixel"); }
}
</SCRIPT>
<IMG ID="ID_Image" SRC="test.jpg">
<SELECT ID="ID_Select" onchange="AusschnittSetzen()">
    <OPTION VALUE=100>100 Pixel </OPTION>
    <OPTION VALUE=40>40 Pixel </OPTION>
    <OPTION VALUE=50>50 Pixel </OPTION>
    <OPTION VALUE=60>60 Pixel </OPTION>
</SELECT>
```

.closed Zustand auf Geschlossenheit eines Fensters
siehe Objekt window

.code Url der *.class-Datei (kompilierter Javacode)

.codeBase Url der Komponente für Download der Komponente vom Server auf den Client
optionale Versionsprüfung möglich (nicht bei Applet-Komponente) um unnötigen
Download zu ersparen: vorheriger Abgleich der Version der Komponente auf Server und Client
auch wenn der Versionsabgleich keinen Download ergab, wird immer HTTP Header-Transaktion
ausgelöst

Beispiel:

```
<OBJECT ID="CommonDialog1"
    WIDTH=32
    HEIGHT=32
    CLASSID="..... "
    CODEBASE="http://www.test.de/test.cab#Version=1,0,0,0"
>
</OBJECT>
```

.codeType Internet Media Type (Mimety) des Codes zum Objekt, das per OBJECT-Tag eingebunden wurde

.color Farbe des Objektes
#rrggbb
vordefinierter Farbname (browserspezifisch)
rgb xyz x für rot, y für grün, z für blau, x bis z reelle Zahl

.colorDepth Anzahl der Bits pro Pixel für eine Farbe UND Verwendung destination device or buffer
wird durch den Wert der Eigenschaft .bufferDepth überschrieben,
wenn .bufferDepth mit Wert > 0
per screen Objekt



<code>.colorDepth</code>	Anzahl der Bits pro Pixel für eine Farbe UND Verwendung destination device or buffer wird durch den Wert der Eigenschaft <code>bufferDepth</code> überschrieben, wenn <code>.bufferDepth</code> mit Wert > 0 Behavior <code>.style.clientCaps</code>				
<code>.cols</code>	Breite aller Frames eines Frameset (Breite des Frameset als Container) Beispiele: <pre><FRAMESET COLS="40%, 60%"> // Summe muss immer 100% sein</pre> <pre><FRAMESET COLS="50, *, 80"> // 50 Pixel, 80 Pixel und restliche Fensterbreite</pre>				
<code>.cols</code>	Anzahl der sichtbaren Zeichen in einer Zeile der TEXTAREA Hinweis: Es können mehr Zeichen in der TEXTAREA enthalten sein als sichtbar siehe <code>.rows</code> für Mehrzeiligkeit <code>.wrap</code> für Wortumbruch Scrollleisten sind erzeugbar siehe <code>textarea</code> Objekt				
<code>.cols</code>	Anzahl der Spalten in der Tabelle wenn belegt so wird Tabelle schneller gerendert siehe Objekt <code>table</code> Beispiel: <pre><TABLE ID="ID_Tabelle" BORDER CELLSPACING=10> <TR> <TD>Zelle 1</TD> <TD>Zelle 2</TD> </TR> </TABLE> <BUTTON onclick="alert(ID_Tabelle.cols);">Anzahl</BUTTON></pre>				
<code>.colSpan</code>	Anzahl der Spalten einer Zelle in einer Tabelle siehe Objekt <code>table.tr.td</code> siehe Objekt <code>table.tr.th</code>				
<code>.Column</code>	aktuelle Spalte des Zeichens im Textstream, also Nummer des Zeichens im Stream nur bei zeichenweiser Dateiverarbeitung verwendbar, das jedes gelesene newline-Zeichen den Wert von <code>.Column</code> auf 1 setzt Beispiel: <pre>var DateiSystem = new ActiveXObject("Scripting.FileSystemObject"); var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0); DateiOffen.WriteLine("Test"); DateiOffen.Close(); DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0); DateiOffen.ReadLine(); alert(DateiOffen.Column); DateiOffen.Close();</pre>				
<code>.compatMode</code>	Kompatibilität des IE 6.x zu CSS1 bzw. seinen Browservorgängern siehe <code>style</code> Objekt und Kodierung von <code>!DOCTYPE</code> ab IE 6.x <code>document.compatMode</code> nur lesen <table> <tr> <td>"BackCompat"</td><td>Kompilationsmodus ist aus IE 6.x Browser unterstützt nur CSS und ist damit kompatibel zu den Browservorgängern</td></tr> <tr> <td>"CSS1Compat"</td><td>Kompilationsmodus ist ein IE 6.x Browser unterstützt nur CSS1 und ist damit nicht kompatibel zu den Browservorgängern</td></tr> </table>	"BackCompat"	Kompilationsmodus ist aus IE 6.x Browser unterstützt nur CSS und ist damit kompatibel zu den Browservorgängern	"CSS1Compat"	Kompilationsmodus ist ein IE 6.x Browser unterstützt nur CSS1 und ist damit nicht kompatibel zu den Browservorgängern
"BackCompat"	Kompilationsmodus ist aus IE 6.x Browser unterstützt nur CSS und ist damit kompatibel zu den Browservorgängern				
"CSS1Compat"	Kompilationsmodus ist ein IE 6.x Browser unterstützt nur CSS1 und ist damit nicht kompatibel zu den Browservorgängern				
<code>.complete</code>	Zustand des Ladens des Objektes				
<code>.connectionType</code>	Typ der genutzten Verbindung bzw. Offline-Status der Verbindung Behavior <code>.style.clientCaps</code>				
<code>.constructor</code>	Bezeichner einer JScript-Klassenkategorie (Objekttyp) oder eines privaten Konstruktors Anwendung: Ermittlung der Objektklasse/Konstruktors eines abgeleiteten Objektes Ableitung aus der Objektklasse per Anweisung <code>new</code> mit der Objektklasse als Konstruktor benötigt (siehe dort) nicht bei Script-Objekt <code>Math</code> möglich Ableitung aus privatem Konstruktor per Anweisung <code>new</code> , die den privaten Konstruktor verwendet JScript-Objektklassen sind z.B.				



Array
Boolean
Date
Function
String

.constructor

Bezeichner einer JScript-Objektklasse (Objekttyp) oder eines privaten Konstruktors

Anwendung: Ermittlung der Objektklasse/Konstruktors eines abgeleiteten Objektes

Ableitung aus der Objektklasse

per Anweisung new mit der Objektklasse als Konstruktor benötigt (siehe dort)

nicht bei Script-Objekt Math möglich

Ableitung aus privatem Konstruktor

per Anweisung new, die den privaten Konstruktor verwendet

JScript-Objektklassen sind z.B.

Array
Boolean
Date
Function

Beispiel 1 für Ableitung aus einem JScript-Objekt:

```
var Kette = new String("Hi");
alert( (Kette.constructor == String));    // liefert "true"
alert( (Kette.constructor == "String"));  // liefert "false"
```

Beispiel 2 für Ableitung anhand privaten Konstruktors:

```
function TestFunktion()
{alert("Hallo");}

var ZeigerAufFunktion = new TestFunktion(); // bewirkt Ausführung von TestFunktion() also auch von alert()

// ZeigerAufFunktion();    // nicht möglich und bringt Fehlermeldung wegen fehlernder Instanz,
                           // da keine Ableitung vom JScript-Objekt Function
                           // Kodierung ohne () bringt keinen Fehler, da als Variablendeklaration erkannt

alert(ZeigerAufFunktion.constructor == TestFunktion);    // true
alert(TestFunktion.constructor == TestFunktion);         // false
```

.content

Wert der Informationen laut Eigenschaften .httpEquiv und .name des meta Objektes

Beispiele:

Dokument alle 2 Sekunden neu laden

<META HTTP-EQUIV="REFRESH" CONTENT=2>

Zeichensatz des Dokumentes

<META HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=Windows-1251">

Themenunterstützung abschalten

<META HTTP-EQUIV="MSTHEMECOMPATIBLE" CONTENT="no">

um sicher zu gehen, daß immer die Seite mit garantiert aktuellem Stand geladen wird

→ eventuell ist Proxyserver nicht aktuell

<META HTTP-EQUIV="expires" CONTENT=sekunden_wert>

sekunden_wert: Wartezeit in Sekunden bis zum nächsten Laden unter Umgehung
des Proxyserver-Cache → 0, also immer umgehen und sofort laden

Suchmaschine beim Scannen des Dokumentes beeinflussen:

Suchmaschine darf alles tun

<META HTTP-EQUIV="Robots" CONTENT="all">

Suchmaschine darf Dokument nichts scannen

<META HTTP-EQUIV="Robots" CONTENT="noindex">

Suchmaschine darf Dokument scannen

<META HTTP-EQUIV="Robots" CONTENT="index">

Suchmaschine darf Verweisen folgen

<META HTTP-EQUIV="Robots" CONTENT="follow">

Suchmaschine darf Verweisen nicht folgen

<META HTTP-EQUIV="Robots" CONTENT="nofollow">

.contentEditable

Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat)



Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
 "inherit" Default. Editierbarkeit von Parent geerbt
 "false" Content nicht veränderbar
 "true" Content ist veränderbar

Beispiel:

```
<HEAD>
<SCRIPT>
function EditierbarkeitWechseln()
{
    var Editierbarkeit = ID_Span1.isContentEditable;
    var Editierbarkeit_Negation = ! Editierbarkeit;
    ID_Span1.contentEditable = Editierbarkeit_Negation;
    ID_Span2.innerText = Editierbarkeit_Negation;

    if (Editierbarkeit_Negation == false)
    { ID_Button.innerText="editierbar machen" }
    else
    { ID_Button.innerText="nicht editierbar machen" }
}
</SCRIPT>
</HEAD>
<BODY onload="ID_Span2.innerText = ID_Span1.isContentEditable;">
    <BUTTON ID="ID_Button" onclick=" EditierbarkeitWechseln()">
        Klick mich
    </BUTTON>
    <SPAN ID="ID_Span1">Diesen Text editieren</SPAN>
    <SPAN ID="ID_Span2">Editierbarkeit</SPAN>
</BODY>
```

.contentWindow Referenz auf das zugehörige Fenster-Objekt laut Collection document.all
 document.all.object.contentWindow

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function LocationAendern()
{
    for(i=0;i<document.all.length;i++)
    {
        if (document.all[i].tagName=="IFRAME")
        { document.all[i].contentWindow.location = "http://www.test.com"; }
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <BUTTON onclick="LocationAendern();">Location aendern</BUTTON>
    <IFRAME SRC="http://www.test.de"></IFRAME>
</BODY>
</HTML>
```

.cookie Cookie des Dokumentes als Stringwert

Beispiel:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
        + "="
        + escape(Value)
        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
```



```

var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

// Feld der Cookies referenzieren
// erzeugt durch Separation anhand Semikolon
var CookieFeld = document.cookie.split("; ");
var AnzahlCookies = CookieFeld.length;

// Feldelement umfasst Name und Wert des Cookie
// Aufbau Cookie_Name = Cookie_Wert
// Separator ist Gleichheitszeichen
// Index 0 für Cookie_Name
// Index 1 für Cookie_Wert
var CookieNameUndWertAlsFeld;

// CookieFeld auslesen und auf Cookie laut Name prüfen
var Gefunden=false;
var Index = 0;
do
{
    // aktuelles Cookie lesen
    CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

    // und auf Name prüfen
    if (Name == CookieNameUndWertAlsFeld [0])
    {
        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

return CookieWert;
}
</SCRIPT>

```

.cookieEnabled Cookiesstatus lesen aber nicht verändern per navigator Objekt
navigator.cookieEnabled
false Cookies sind nicht erlaubt
true Cookies sind erlaubt

.cookieEnabled Cookienutzbarkeit im Browser
Behavior .style.clientCaps
false Browser unterstützt keine Cookies
true Browser unterstützt Cookies

.cookieEnabled Cookienutzbarkeit im Browser
siehe Objekt window.clientInformation

.coords Koordinaten eines Objektes in Abhängigkeit zur Eigenschaft .shape

Beispiel 1:

```

<IMG SRC="test.gif" USEMAP="#MapName">
<MAP NAME="#MapName">
    <AREA SHAPE="rect"      COORDS="0,0,33,33" HREF="test.gif">
    <AREA SHAPE="circle"      COORDS="90,33,3" HREF="test1.gif">
</MAP>

```

Beispiel 2:

```

<MAP NAME="logischer_map_name"
<AREA
    NAME="name_des_verweissensitiven_bereiches_der_grafik"
    COORDS="koordinaten_liste_des_bereiches"
    HREF="url"
    SHAPE= "rect"
           oder "poly"
           oder "circle"
           oder "default"
    TARGET="logischer_window_name"
    onClick="eventhandler_1"
    onMouseOut="eventhandler_2"
    onMouseOver="eventhandler_3"

```



```

> .....
</MAP>

SHAPE:      rect      Rechteck
            poly      Vieleck
            circle    Kreis
COORDS: bei rect und poly "x1,y1,....,xn,yn"
            circle      "x,y,r"
            x           Spalte in Pixel
            y           Zeile in Pixel
            r           Radius in Pixel
Hinweis: mehrere AREA sind möglich

```

Beispiel 3:

```

<IMG SRC="test.gif" WIDTH=504 HEIGHT=126 BORDER=0 ALT="Gesamtes Bild"
USEMAP="# freier_mappen_name"
>
<MAP NAME="freier_mappen_name">
<AREA SHAPE="rect"
COORDS="0,0,82,126"
ALT="Teilbereich 1"
HREF="/graphics/test.gif"
STYLE="....."
onxxx="....."
>
.....
<AREA ....>
</MAP>

```

.copyright

Copyright der Media-Datei auf der Timeline
bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von COPYRIGHT des aktiven
Eintrages geliefert und nicht den der Datei.
Track entspricht playItem Objekt laut object.playlist.item() bzw. object.playlist.aktiveTrack
Trackliste liegt in der Behavior-Collection .style.time2.playlist als Playliste:
Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```

var Kette = object.playlist.item(index).copyright;

            index      Integer, ab 0

var CopyrightVariable = object.playlist.aktiveTrack.copyright;

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:VIDEO          ID="ID_Video"
                  SRC="test.wmv"
                  STYLE="position:absolute;top:50px;height:100px"
>
</t:VIDEO>
<SPAN ID="ID_Span"
      STYLE="position:absolute;top:165px;"
>
</SPAN>
<BUTTON ID="ID_Button"
        onclick="ID_Span.innerText= ID_Video.copyright"
>
      Klick
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">

```



```

<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
        ID_Span2.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"
        TIMEACTION="visibility"
        onend="ButtonUpdate();"
        ontrackchange="AnzeigeUpdate();"
        onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>
    <BR>

```



```

<SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

<BR>
<BUTTON ID="ID_Button1"
onclick="ID_Media.beginElement(); ButtonUpdate();"
>
    Start
</BUTTON>
<BUTTON ID="ID_Button2"
onclick="ID_Media.playlist.nextTrack();"
>
    naechster Track
</BUTTON>
<BUTTON ID="ID_Button3"
onclick="ID_Media.playlist.prevTrack();"
>
    vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
onclick="ID_Media.endElement(); AnzeigeLoeschen();"
>
    Stop
</BUTTON>
</BODY>
</HTML>

```

.Count Anzahl der Elemente im Dictionary

.Count Anzahl der Elemente in der Collection FileSystemObject.Drives

.Count Anzahl der Elemente in der Collection FileSystemObject.Folder.Files

.Count Anzahl der Elemente in der Collection FileSystemObject.Folder.Folders

.cpuClass CPU-Klasse per navigator Objekt
navigator.cpuClass
"x86" Intel
"68K" Motorola
"Alpha" Digital
"PPC" Motorola
"Other" alles andere, auch Sun SPARC

.cpuClass CPU-Hersteller
Behavior .style.clientCaps
"x86" Intel
"68K" Motorola
"Alpha" Digital
"PPC" Motorola
"Other" alles andere, auch Sun SPARC

.cpuClass CPU-Hersteller
siehe Objekt window.clientInformation
"x86" Intel
"68K" Motorola
"Alpha" Digital
"PPC" Motorola
"Other" alles andere, auch Sun SPARC

.cssText Style-Attribut-Wert (Style-Regel, CSS-Attributwert)
ab IE 5.x
Hinweis: attributes Collection referenziert nicht Style-Attribute !

Beispiel:

```

<P ID="ID_P" STYLE="color:'green'; font-weight:bold">Test-Text</P>
<BUTTON onclick="alert(ID_P.style.cssText)"> CSS-Attributwerte anzeigen</BUTTON>

```

.ctrlKey CTRL-Tasten-Status
Objekt event
false CTRL-Taste ist nicht gedrückt
true CTRL-Taste ist gedrückt

Beispiel:

```

<HEAD>
<SCRIPT>
    function InnerTextSetzen(Zeiger, Kette)
    {Zeiger.innerText=Kette;}

```




```

function init()
{
    InnerTextSetzen(ID_Span1,'false');
    InnerTextSetzen(ID_Span2,'false');
}
function CtrlDown()
{
    if (event.ctrlLeft)
    { InnerTextSetzen(ID_Span1,'true'); }
    else
    {
        if (event.ctrlKey)
        { InnerTextSetzen(ID_Span2,'true'); }
    }
}

function CtrlUp()
{
    if (!event.ctrlKey)
    {
        InnerTextSetzen(ID_Span1,'false');
        InnerTextSetzen(ID_Span2,'false');
    }
}
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="CtrlDown();" onkeyup="CtrlUp();">
    <TABLE>
        <TR>
            <TD><I>Linke CTRL-Taste gedrueckt</I></TD>
            <TD><I>Rechte CTRL-Taste gedrueckt</I></TD>
        </TR>
        <TR>
            <TD ALIGN="center">
                <SPAN ID="ID_Span1"></SPAN>
            </TD>
            <TD ALIGN="center">
                <SPAN ID="ID_Span2"></SPAN>
            </TD>
        </TR>
    </TABLE>
</BODY>

```

.ctrlLeft

linken CTRL-Taste Status
 Objekt event
 nicht für Win9x
 nur für Dokument, das den Fokus hat
 false linke CTRL-Taste ist nicht gedrückt
 true rechte CTRL-Taste ist gedrückt

Beispiel:

```

<HEAD>
<SCRIPT>
function InnerTextSetzen(Zeiger, Kette)
{ Zeiger.innerText=Kette; }

function init()
{
    InnerTextSetzen(ID_Span1,'false');
    InnerTextSetzen(ID_Span2,'false');
}
function CtrlDown()
{
    if (event.ctrlLeft)
    { InnerTextSetzen(ID_Span1,'true'); }
    else
    {
        if (event.ctrlKey)
        { InnerTextSetzen(ID_Span2,'true'); }
    }
}

function CtrlUp()
{

```



```

        if (!event.ctrlKey)
        {
            InnerTextSetzen(ID_Span1,'false');
            InnerTextSetzen(ID_Span2,'false');
        }
    }
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="CtrlDown();" onkeyup="CtrlUp();">
    <TABLE>
        <TR>
            <TD><I>Linke CTRL-Taste gedrueckt</I></TD>
            <TD><I>Recchte CTRL-Taste gedrueckt</I></TD>
        </TR>
        <TR>
            <TD ALIGN="center">
                <SPAN ID="ID_Span1"></SPAN>
            </TD>
            <TD ALIGN="center">
                <SPAN ID="ID_Span2"></SPAN>
            </TD>
        </TR>
    </TABLE>
</BODY>

```

.currentFrame Nummer des aktuellen Frame einer Media-Datei auf der Timeline
nicht alle Media-Typen unterstützen Frames
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Media"
            SRC="test.avi"
            STYLE="width:175px; height:150px;"
    >
    </t:VIDEO>
    <BR>
    <SPAN ID="ID_Span"
        CLASS="time_line_Klasse"
        DUR="0.01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Media.currentFrame);"
    >
    </SPAN>
    <BR>
    <BUTTON ID="ID_Button"
        onclick="ID_Media.beginElement();"
    >
        Restart
    </BUTTON>
</BODY>
</HTML>

```

.currentItem Zeiger auf aktuellen Media-Eintrag (Objekt MediaItem)
Eintrag stammt aus der Playliste, wenn diese existent ist, und ist der aktuelle Eintrag der Playliste
siehe Behavior .style.mediaBar

.data Url der Daten des Objektes

.dataFld Datenquelle-Name vergeben (ID)

Beispiel 1:

```

<TABLE DATASRC="#ice_cream">
    <TR>
        <TD>
            <INPUT TYPE=TEXT DATAFLD="flavor">
        </TD>
    </TR>

```



</TABLE>

Beispiel 2:

```
<SELECT DATASRC="#Anker" DATAFLD="KartenTyp">
  <OPTION>Visa
  <OPTION>Mastercard
  <OPTION>American Express
  <OPTION>Diner's Club
  <OPTION>Sparkasse
</SELECT>
```

.dataFormatAs	Datenquelle-Anzeigeart
"text"	Default. Data als Text anzeigen
"html"	Data als HTML anzeigen
"localized-text"	ab IE 5.01 Data in den lokalen Einstellungen des Clients anzeigen

Beispiel:

```
<DIV DATAFLD="Column2" DATAFORMATAS="html"></DIV>
```

.dataPageSize	Anzahl der sichtbaren Datensätze auf einer Tabellenseite
	Anzahl der Sätze pro Dataset-Anzeige
	siehe Objekt table

Beispiel:

Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815 Willmann;Theo;1234 Xantippe;Isa;5678 Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
  var SortierRichtung = true;

  function Sortieren(FeldBezeichner)
  {
    // Sortierrichtung festlegen
    var Kette = "-"; // Annahme
    if (SortierRichtung) { Kette = "+"; }

    // nächste Sortierung umgekehrt
    SortierRichtung = !SortierRichtung;

    // sortieren
    ID_Datenbank.Sort= Kette + FeldBezeichner;

    // und das Ergebnis anzeigen
    ID_Datenbank.Reset();
  }

  function vorwaerts(AnzahlSaetze)
  {
    // nächste Seite anzeigen
    document.all.ID_Tabelle.nextPage();

    // und Satzzeiger korrigieren
    for (var i = 0; i < AnzahlSaetze; i++)
    {
      if (ID_Datenbank.recordset.AbsolutePosition !=
          ID_Datenbank.recordset.RecordCount)
      { ID_Datenbank.recordset.MoveNext(); }
    }

    if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
    { alert("Letzter Datensatz erreicht!"); }
  }

  function rueckwaerts(AnzahlSaetze)
  {
    // vorhergehende Seite anzeigen
```



```

        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == 1)
        {alert("Erster Datensatz erreicht!");}
    }

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL"    VALUE="adress.txt">
    <PARAM NAME="UseHeader"  VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
>
    <THEAD>
        <TR>
            <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
            <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
            <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD><DIV DATAFLD="vorname"></DIV></TD>
            <TD><DIV DATAFLD="name"></DIV></TD>
            <TD><DIV DATAFLD="telefon"></DIV></TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT
    TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
    <!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT
    TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
    <!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->
</BODY>
</HTML>

```

.dataSrc Datenquelle als Anker festlegen

Beispiel:

```

<TABLE DATASRC="#anker">
    <TR>
        <TD>
            <INPUT TYPE=TEXT DATAFLD="customer_name">
        <TD>
    </TR>
</TABLE>

```

.DateCreated Datum und Zeit der Ordnererstellung liefern

Beispiel:

```

var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner           = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateCreated);

```

.DateCreated Datum und Zeit der Dateierstellung liefern



Beispiel:

```
var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateCreated);
```

.DateLastAccessed Datum und Zeit des letzten Ordnerzugriffes liefern

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateLastAccessed);
```

.DateLastAccessed Datum und Zeit des letzten Dateizugriffes liefern

Beispiel:

```
var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateLastAccessed);
```

.DateLastModified Datum und Zeit der letzten Ordneränderung liefern

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.DateLastModified);
```

.DateLastModified Datum und Zeit der letzten Dateiänderung liefern

Beispiel:

```
var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateiNameMitPfad);
alert(Datei.DateLastModified);
```

.decelerate

Verlangsamung des Elementes auf der Timeline
 hat keinen Einfluss auf die Dauer der Timeline
 auch wirksam bei Wiederholungen per Attribute `.repeatCount` oder `.repeatDur`
 Summe der Werte der Attribute `.accelerate` und `.decelerate` darf nicht 1 überschreiten
 wenn ja, so werden beide Attribute ignoriert, also nicht verwendet
 siehe Objekt `currTimeState` und Behavior `.style.time2`
 siehe `.accelerate`

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<DIV ID="ID_Div" CLASS="time_line_klasse" STYLE=".....">
</DIV>
<t:ANIMATE TARGETELEMENT="ID_Div"
ATTRIBUTENAME="left"
TO="400"
DUR="3"
decelerate="1"
REPEATCOUNT="3"
>
</t:ANIMATE>
</BODY>
</HTML>
```

.declare Zeichenkette für Declare-Funktionalität des Objektes, das per OBJECT-Tag eingebunden wurde

.defaultCharset regionaler Standardzeichensatz (Charakter-Set) zum Encoden eines Objektes im Dokument

.defaultChecked Selektionsstatus des Checkbox-Control bzw. Radio Button-Control bzw. Radio Button-Control
 bezüglich Standard-Selektion
 (Objekt `input checkbox` bzw. Objekt `input radio`)
 wird bei Checkbox-Control auch verändert durch Eigenschaft `.indeterminate`
 true Default



	Control-Element hat Standardselektion
false	Control-Element hat keine Standardselektion
.defaultSelected	Default-Selektionsstatus der Option, also Objektes document.select.option des Objektes document.select standardgemäß ist immer die oberste Option selektiert es kann aus einer Optionengruppe immer nur genau 1 Option selektiert sein
true	Default Option ist selektiert
false	Option ist nicht selektiert
.defaultStatus	Standard-Text der Statuszeile (nicht der aktuelle Text) siehe .status Verwendung in Eventhandler-Funktion: Es muss return true; kodiert werden siehe Objekt window
.defaultValue	Standardwert laut Objektinitialisierung für Element innerhalb eines Formulars z.B. Reset des Formular löst Initialisierung des Elementes aus

Beispiel für INPUT-Objekt und dessen Varianten:

Wert ist immer String

für Input-Elemente (input Objekt) gelten folgende Standardwerte:

INPUT type=checkbox	on
INPUT type=reset	Reset
INPUT type=submit	Submit Query

alle anderen Input-Elemente haben keinen Standardwert

Hinweis: sendbar sind folgende Werte:

INPUT type=checkbox	selektierter Wert
INPUT type=file	Dateiname laut Eingabe
INPUT type=hidden	selektierter Wert einer Box-Control (selektierte Option)
INPUT type=password	Eingabewert
INPUT type=radio	selektierter Wert
INPUT type=reset	das Label des Elementes (falls Label existent ist)
INPUT type=submit	das Label des Elementes (falls Label existent ist)
INPUT type=text	Eingabewerte

Werte können gelesen und geschrieben werden
nur lesen bei INPUT type=file

.defaultValue	Vorbelegung der TEXTAREA sichtbare Auswirkungen nur bei Instanzierung des Objektes Formular-Reset siehe textarea Objekt
.defer	Pars-Status des Scriptes per script Objekt download eines Scriptes kann beschleunigt werden, wenn es vom Parsen zurückgestellt wird
.description	Beschreibung des Run-Time-Errors deprecated: dafür Eigenschaft .message verwenden, die indentische Funktion hat siehe error JScript-Objekt
.designMode	Editierbarkeit des Dokumentes (Manipulierbarkeit) z.B. per Ausführung von Javascript
.deviceXDPI	Aktuelle Anzahl der horizontalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm per screen Objekt

Beispiel 1:

```
<SCRIPT>
function ScaleFactorX()
{
    var ScaleFactor = screen.deviceXDPI / screen.logicalXDPI;
    return ScaleFactor;
}
</SCRIPT>
```

Beispiel 2:

```
<SCRIPT>
function fnScaleManually()
{
    // normale DPI-Auflösung (Dot per Inch)
    var constNorm = 96;
```



```

// Zoomen
if ( (screen.deviceXDPI == screen.logicalXDPI)
    && (screen.deviceXDPI > constNorm)
    )
{
    document.body.style.zoom = constNorm / screen.logicalXDPI;
}
}
</SCRIPT>

```

.deviceYDPI Aktuelle Anzahl der vertikalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm per screen Objekt

Beispiel 1:

```

<SCRIPT>
function ScaleFactorY()
{
    var ScaleFactor = screen.deviceXDPI / screen.logicalYDPI;
    return ScaleFactor;
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function fnScaleManually()
{
    // normale DPI-Auflösung (Dot per Inch)
    var constNorm = 96;

    // Zoomen
    if ( (screen.deviceYDPI == screen.logicalYDPI)
        && (screen.deviceYDPI > constNorm)
        )
    {
        document.body.style.zoom = constNorm / screen.logicalYDPI;
    }
}
</SCRIPT>

```

.dialogArguments Zeiger auf Objekt window.dialogArguments als Argumente für modale oder nicht modale Dialoge ab IE 5.x
wird per.showModalDialog() bzw .showModelessDialog() instanziiert
siehe Objekt window

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function Anzeige()
{
    // Zeiger auf die Formularelemente holen
    var FormularElemente = ID_Formular.elements;

    // Formulardaten in einem privaten Objekt kapseln als Argument für modalen Dialog
    var FormularDaten = new Object();
    FormularDaten.firstName = FormularElemente.ID_Input1.value;
    FormularDaten.lastName = FormularElemente.ID_Input2.value;

    // Fenster als modalen Dialog anzeigen und dabei Eigenschaft .dialogArguments füllen
    // Mit der Übergabe der Daten erfolgt das Öffnen des neuen Fenster, das
    // sich auf die namensgleichen Eigenschaften von FormularDaten
    // beruft, deren Bezeichner mit in den Argumenten übergeben werden
    window.showModalDialog( "test.htm",
        FormularDaten,
        "dialogHeight:300px; dialogLeft:200px;"
    );
}
</SCRIPT>
</HEAD>
<BODY>
<FORM ID= "ID_Formular">
    Vorname:
    <INPUT TYPE="text" NAME="ID_Input1" VALUE="Vorname">
    <BR>
    Nachname:
    <INPUT TYPE="text" NAME="ID_Input2" VALUE="Nachname">

```




```
</FORM>
<BR>
<BUTTON onclick="Anzeige();" >Formulardaten im modalen Dialog anzeigen</BUTTON>
</BODY>
</HTML>
```

test.htm enthält:

```
<HTML>
<HEAD>
<SCRIPT>

    function Anzeigen()
    {
        var DialogArgumente = window.dialogArguments;

        // Es müssen namensidentische Bezeichner der Eigenschaften von
        //     DialogArgumente verwendet werden:
        //         firstName und lastName werden in der
        //             aufrufenden Webseite definiert
        //             und müssen hier ebenfalls verwendet werden
        document.writeln("Vorname = " + DialogArgumente.firstName);
        document.write("Nachname = " + DialogArgumente.lastName);
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeigen();">
</BODY>
</HTML>
```

<code>.dialogHeight</code>	Höhe des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
	ab IE 5.x siehe Objekt <code>window</code>

Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>
```

<code>.dialogLeft</code>	X-Koordinate der linken oberen Ecke des Dialogfensters, das mit Methoden <code>.showModalDialog()</code> bzw. <code>.showModelessDialog()</code> erzeugt wurde
	ab IE 5.x
	siehe Objekt <code>window</code>

Beispiel:

```
<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
```



```

        <SELECT onchange="ZeigeModalenDialog()">
            <OPTION>Eintrag 1</OPTION>
            <OPTION>Eintrag 2</OPTION>
            <OPTION>Eintrag 3</OPTION>
        </SELECT>
    </BODY>

```

.dialogTop Y-Koordinate der linken oberen Ecke des Dialogfensters, das mit Methoden `.showModalDialog()` bzw. `.showModelessDialog()` erzeugt wurde
ab IE 5.x
siehe Objekt window

Beispiel:

```

<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>

```

.dialogWidth Breite des Dialogfensters, das mit Methoden `.showModalDialog()` bzw. `.showModelessDialog()` erzeugt wurde
ab IE 5.x
siehe Objekt window

Beispiel:

```

<HEAD>
<SCRIPT>
function ZeigeModalenDialog()
{
    event.srcElement.blur();                // Focus dem QuellElement wegnehmen
    window.showModalDialog("test.htm",
        "",
        "dialogWidth:5cm; dialogHeight:10cm; dialogTop:0cm; dialogLeft:0cm"
    );
}
</SCRIPT>
</HEAD>
<BODY>
    <SELECT onchange="ZeigeModalenDialog()">
        <OPTION>Eintrag 1</OPTION>
        <OPTION>Eintrag 2</OPTION>
        <OPTION>Eintrag 3</OPTION>
    </SELECT>
</BODY>

```

.dir Umflussrichtung
"ltr" Default.
Umfluss von links nach rechts
"rtl" Umfluss von rechts nach links

.direction Start-Scrollrichtung des marquee Objektes
"left" Default
Scrollrichtung links
"right" Scrollrichtung rechts
"down" Scrollrichtung runter
"up" Scrollrichtung rauf

.disabled Interaktionsfähigkeit
nur wenn sichtbar so User-Interaktion möglich
true Element nicht interaktionsfähig



Beispiel 1: false Default, Element ist interaktionfähig

```
<STYLE ID="ID_Style">
.styletest{background-color: black; color: white;}
</STYLE>
<SCRIPT>
function Wechsel()
{
    if(ID_P.enablement == "enabled")
    {
        ID_P.enablement = "disabled";
        ID_Button2.value = "unsichtbar";
        ID_Style.disabled = true;
        ID_Button1.disabled = true;
    }
    else
    {
        ID_P.enablement = "enabled";
        ID_Button2.value = "sichtbar";
        ID_Style.disabled = false;
        ID_Button1.disabled = false;
    }
}
</SCRIPT>
<P ENABLEMENT="enabled" ID="ID_P" CLASS="styletest">
    Test Text
    <INPUT TYPE="button" ID="ID_Button1" CLASS="styletest"
        VALUE="Testen" onclick="alert('Test')"
    >
</P>
<INPUT TYPE="button" ID="ID_Button2" VALUE="Wechseln"
    onclick="Wechsel()"
>
```

Beispiel 2 für Sekundenbalken:

```
<HTML>
<HEAD>
<SCRIPT>
// ++++++ globale Variablen, die verändert werden können
var SoundUrl = "56sec.mid";
var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++ Routinen der Sekundenzählung
var SekundenZahler = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Ausdrücke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob SekundenZählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}
```



```

    }

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekunden zählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl,SoundFileDauerInSekunden)
{
    this.SoundFileUrl = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString()+ " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ----- Variablen init
    SekundenZahler = 0;
    SekundenZahlerTimeoutID = null;

    // ----- visuelle Anzeige erzeugen

```



```

// - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
//      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
//      Der Ausdruck liefert den Wert , welcher sofort das Layout der
//      DIV's beeinflusst.
//      Jeder Ausdruck besitzt den SekundenZähler als Komponente.
//      Damit ändert sich der Wert des Ausdruckes.
//      Für die Neuberechnung des Ausdruckes ist der Aufruf von
//      document.recal()
//      nötig.
//      Dieser Aufruf erfolgt in SekundenZahlen(), also permanent pro Sekunde.
//      Damit wird der Style-Wert permanent neu berechnet.
//      Damit visualisieren sich die DIV's permanent neu.

// Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
//      also dynamisch anzeigen
ID_DIV_Balken.style.setExpression( "width",
                                   "SekundenZähler * PixelBreiteProBalkenErweiterung"
                                   );

// Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
//      also dynamisch anzeigen
ID_DIV_SekundenZähler.setExpression("innerText","SekundenZähler.toString()");

// - - - Messlatte statisch anzeigen
ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                               + SoundDauerInSekunden.toString()
                               + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + 'STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'
                    );

    document.write( ' <DIV ID="ID_DIV_SekundenZähler"'
                    + 'STYLE="color:hotpink;font-weight:bold"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'
                    );

    document.write( ' <DIV ID="ID_DIV_MessLatte"'
                    + 'STYLE="color:white;background-color:gray"'
                    + '>'
                    + '</DIV>'
                    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ---- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ---- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
<!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

.disabledUI

Sichtbarkeit des User-Interfaces der Media Bar
siehe Behavior .style.mediaBar



Beispiel:

```
<DIV ID="ID_Div"
  STYLE="behavior:url(#default#mediaBar)"
>
</DIV>
<INPUT TYPE=button
  VALUE='abspielen von test.asx'
  onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
            ID_Div.disabledUI = true;
            "
>
```

.doctype	Referenz auf Dokumenttyp des Dokumentes document.doctype
.document	Zeiger auf das Objekt document im Fenster siehe Objekt window
.document	Zeiger auf das HTML-Dokument im Popup-Fenster, das per .show() instanziiert wird Es können damit alle Eigenschaften und Methoden des Objektes document referenziert werden siehe Objekt window.popup

Beispiel für diverse Meldungsfenster per **jeweiligem** PopUp-Fenster (es kann immer nur genau 1 Popup-Fenster angezeigt werden):

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">

    var PopUpFensterFeld = new Array();

    // nachfolgende Felder beschreiben die PopUp-Fenster und müssen
    // identische Anzahl der Feldelemente haben
    // Feld-Index ist die Nummer des PopUp-Fensters

    var PopUpFenster_RahmenStyle_Feld = new Array
    (
        "solid black 4px",
        "solid blue 3px",
        "solid green 2px"
    );

    var PopUpFenster_HintergrundFarbe_Feld = new Array
    (
        "yellow",
        "gray",
        "orange"
    );

    var PopUpFenster_Inhalt_Feld = new Array
    (
        "<B>Inhalt PopUpFenster 0<B>",
        "Inhalt PopUpFenster 1",
        "<TT>Inhalt PopUpFenster 2<TT>"
    );

    var PopUpFenster_X_Koordinate_Feld = new Array
    (
        100,
        150,
        200
    );

    var PopUpFenster_Y_Koordinate_Feld = new Array
    (
        150,
        200,
        250
    );

    var PopUpFenster_Breite_Koordinate_Feld = new Array
    (
        80,
        140,
```



```

        200
    );

    var PopUpFenster_Hoehe_Koordinate_Feld = new Array
    (
        100,
        140,
        180
    );

    var AnzahlPopUpFenster = PopUpFenster_Hoehe_Koordinate_Feld.length;

    function PopupFensterInstanzieren(NummerDesPopUpFensters, ZeigerAufElternFenster)
        // NummerDesPopUpFensters ab 0
    {
        PopUpFensterFeld[NummerDesPopUpFensters] =
            ZeigerAufElternFenster.createPopup();
    }

    function PopupFensterFuellen(NummerDesPopUpFensters)
    {
        // Body des Popup-Fensters gestalten
        var PopupFenster_Body =
            PopUpFensterFeld[NummerDesPopUpFensters].document.body;

        PopupFenster_Body.style.backgroundColor =
            PopUpFenster_HintergrundFarbe_Feld[NummerDesPopUpFensters];

        PopupFenster_Body.style.border =
            PopUpFenster_RahmenStyle_Feld[NummerDesPopUpFensters];

        PopupFenster_Body.innerHTML =
            PopUpFenster_Inhalt_Feld[NummerDesPopUpFensters];
    }

    function PopupFensterAnzeigen(NummerDesPopUpFensters,
        ObjektZuDemPopUpFensterRelativPositioniertIst
    )
    {
        // Popup-Fenster anzeigen und damit öffnen
        PopUpFensterFeld[NummerDesPopUpFensters].show(
            PopUpFenster_X_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Y_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Breite_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Hoehe_Koordinate_Feld[NummerDesPopUpFensters],
            ObjektZuDemPopUpFensterRelativPositioniertIst
        );
    }

    function PopupFensterSchliessen(NummerDesPopUpFensters)
    {
        var Zeiger = PopUpFensterFeld[NummerDesPopUpFensters];

        if (Zeiger.isOpen)
        { Zeiger.hide(); }
    }

    function Init()
    {
        // PopupFenster instanzieren im aktuellen Fenster (window)
        for (var i = 0 ; i < AnzahlPopUpFenster; i++)
        {
            PopupFensterInstanzieren(i, window);
            PopupFensterFuellen(i);
        }
    }
</SCRIPT>
</HEAD>
<BODY onload="Init();">
    Durch Klick ausserhalb des Popup-Fensters wird dieses auch automatisch geschlossen.
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 anzeigen"

```




```

        onclick=" PopupFensterAnzeigen(0, document.body);"
    >
    <INPUT  TYPE=button
        VALUE="PopUpFenster 1 anzeigen"
        onclick=" PopupFensterAnzeigen(1, document.body);"
    >
    <INPUT  TYPE=button
        VALUE="PopUpFenster 2 anzeigen"
        onclick=" PopupFensterAnzeigen(2, document.body);"
    >
    <BR>
    <INPUT  TYPE=button
        VALUE="PopUpFenster 0 schliessen"
        onclick=" PopUpFensterSchliessen(0);"
    >
    <INPUT  TYPE=button
        VALUE="PopUpFenster 1 schliessen"
        onclick=" PopUpFensterSchliessen(1);"
    >
    <INPUT  TYPE=button
        VALUE="PopUpFenster 2 schliessen"
        onclick=" PopUpFensterSchliessen(2);"
    >
</BODY>
</HTML>

```

.documentElement Referenz auf Wurzelknoten (root node) des Dokumentes liefern
 Beispiel:

```

<SCRIPT>
    function HoleHTML()
    {ID_Textarea.value= document.documentElement.innerHTML;}
</SCRIPT>
<TEXTAREA ID="ID_Textarea" COLS = 50 ROWS = 10>
</TEXTAREA>

```

.domain Domain-Suffix des Dokumentes
 Kommunikation von Dokumenten verschiedener Urls mit gemeinsamen Domainsuffix
 document.domain

Beispiel:

home.test.com und **www.test.com** können kommunizieren,
 wenn Kette auf **"test.com"** gesetzt wurde in den Dokumenten beider Urls

.downloadCurrent Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei auf der Timeline
 (Laden des Daten-Stream in Form der Media-Datei)
 nur für Media-Datei mit Datenfluss
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:MEDIA          ID="ID_Media"
                     SRC="test.asx"
                     SYNVBHAVIOR="locked"
                     TIMEACTION="display"
    >
    </t:MEDIA>
    <SPAN  ID="ID_Span"
        CLASS="time_line_klasse"
        DUR="0.1"
        REPEATCOUNT="indefinite"
        onrepeat="ID_Span.innerText= ID_Media.downloadCurrent;"
    >
        0
    </SPAN>
</BODY>
</HTML>

```

.downloadTotal Anzahl der insgesamt downgeloadeten Bytes nach dem kompletten Laden einer Media-Datei
 auf der Timeline



(Laden des Daten-Stream in Form der Media-Datei)
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:MEDIA ID="ID_Media"
SRC="test.asx"
SYNVBEHAVIOR="locked"
TIMEACTION="display"
>
</t:MEDIA>
<SPAN ID="ID_Span"
CLASS="time_line_klasse"
DUR="0.1"
REPEATCOUNT="indefinite"
onrepeat="ID_Span.innerText= ID_Media.downloadTotal;"
>
0
</SPAN>
</BODY>
</HTML>
```

.Drive Buchstaben des Laufwerkes liefern, auf dem der Ordner liegt

Beispiel:

```
var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Drive);
```

.Drive Buchstaben des Laufwerkes liefern, auf dem die Datei liegt

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.Drive);
```

.DriveLetter Laufwerksbuchstaben liefern (ohne : und \\ etc.)

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Buchstabe = Laufwerk.DriveLetter;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Buchstabe);
```

.DriveType Laufwerkstyp liefern

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Typ = Laufwerk.DriveType;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Typ);
```

.dropEffect Cursor-Layout bei Drop
 wird von Eigenschaft .effectAllowed beeinflusst
 nur Objekt event.dataTransfer
 "copy" Copy-Cursor wird angezeigt
 "link" Link-Cursor wird angezeigt
 "move" Move-Cursor wird angezeigt
 "none" Default, Standard-Cursor wird angezeigt

Beispiel für Clipboard-Nutzung mit Drag&Drop:

```
<HTML>
<HEAD>
<SCRIPT>
// ##### Quellobjekt #####
```



```

function EventHandlerFuerOnDragStart()
    // Quellobjekt löst Event aus:
    //      in Quelle wird markiert und selektiert
{
    // selektierte Quell-Daten in die Zwischenablage puffern
    //      (Puffer wird per window.event-Objekt verwaltet)
    //      Datentyp ist hier uninteressant, da für die Zwischenablage
    //      der Typ automatisch erkannt wird, also
    //      Speicherung der Daten in der Zwischenablage
    //      immer typgerecht ist
    var ZwischenAblage = window.event.dataTransfer;

    // und diese Daten als verschiebbar erklären:
    //      aus der Quelle in die Zwischenablage
    ZwischenAblage.effectAllowed = "move";
}

// ##### Zielobjekt #####
function EventHandlerFuerOnDragEnter
    // Zielobjekt löst Event aus
    // Maus über Ziel nach dem Draggen der Daten,
    //      UND Maustaste ist noch nicht losgelassen
    // also Zielobjekt wird mit den Daten betreten
{
    // Daten adressieren
    var ZwischenAblage = window.event.dataTransfer;

    // und diese Daten als verschiebbar erklären:
    //      aus der Zwischenablage in das Zielobjekt
    ZwischenAblage.dropEffect = "move";

    // Event-Handler-Rückkehrcode
    var EventObjekt = window.event;
    EventObjekt.returnValue = false;
}

function EventHandlerFuerOnDrop
    // Zielobjekt löst Event aus
    // Maus über Ziel nach dem Droppen der Daten,
    //      UND Maustaste wird losgelassen
{
    // Ziel adressieren, das mit ondrop-Ereignis behandelt werden soll
    var Ziel = window.event.srcElement;

    // Daten in der Zwischenablage adressieren und holen
    //      (Puffer wird per window.event-Objekt verwaltet)
    var ZwischenAblage = window.event.dataTransfer;

    // und Daten vom Texttyp aus der Zwischenablage im Ziel ablegen,
    //      da Ziel nur Textdaten empfangen kann
    Ziel.innerText += Daten.getData("text");

    // Event-Handler-Rückkehrcode
    var EventObjekt = window.event;
    EventObjekt.returnValue = false;
}

function EventHandlerFuerOnDragOver
    // Zielobjekt löst Event aus
{
    // nichts tun ausser Rückkehrcode des Handlers liefern
    var EventObjekt = window.event;
    EventObjekt.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY>
    <DIV ondragstart=" EventHandlerFuerOnDragStart ()"
    >
        Diesen Text markieren und draggen
    </DIV>
    <BR>
    <DIV
        ondragover="EventHandlerFuerOnDragOver()"
        ondragenter="EventHandlerFuerOnDragEnter()"

```



```

ondrop="EventHandlerFuerOnDrop()"
>
An diese Stelle den Text dropfen
</DIV>
</BODY>
</HTML>

```

.dur

Dauer Objektaktivitäten laut Eigenschaft .timeAction
alternativ: Eigenschaft .end
siehe Objekt currTimeState und Behavior .style.time2
Wert im Time-Format des Behavior

z.B.	"h:min:s.f"
nicht	id.event
nicht	id.event+zeit_wert_als_string

Beispiele

"25:45:10"	25 Stunden, 45 Minuten, 10 Sekunden
"45:35"	45 Minuten, 35 Sekunden
"45:00.275"	45 Minuten, 0,275 Sekunde
"10.5"	10,5 Sekunden

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter"
TYPE="fade"
DUR="8"
END="4"
TARGETELEMENT="ID_Div"
>
</t:TRANSITIONFILTER>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
DUR="indefinite"
STYLE="background-image:url(test.gif); background-repeat: no-repeat;"
>
</DIV>
</BODY>

```

.dur

zeitliche Gesamtdauer des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt
siehe Objekt currTimeState und Behavior .style.time2
Timeformat des time2-Behavior

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter"
FROM="0.3"
BY="0.4"
TYPE="barWipe"
DUR="3"
TARGETELEMENT="ID_Div"
>
</t:TRANSITIONFILTER>
<DIV ID="ID_Div"
CLASS="time_line_Klasse"
DUR="indefinite"
STYLE="position:relative; left:20px; width:420px; height:100px;
background-image:url(test.gif); background-repeat: no-repeat;"
>
</DIV>
</BODY>
</HTML>

```



.duration	Dauer des MediaItem Objektes in Sekunden siehe Behavior .style.mediaBar
.dynsrc	Adresse von Videoclip oder VRML Hinweis: für statisches Bild bitte Eigenschaft .src verwenden !!
.E	eulersche Zahl siehe Script-Objekt Math
.effectAllowed	Art von Drag und Drop beeinflusst Eigenschaft .dropEffect nur Objekt event.dataTransfer "copy" kopieren "link" verlinken "move" verschieben "copyLink" kopieren bzw. verlinken, je nach dem was Ziel zulässt "copyMove" kopieren bzw. verschieben, je nachdem was Ziel zulässt "linkMove" verschieben bzw. verlinken, je nachdem was Ziel zulässt "all" alle Arten "none" Ziel kann kein Dropping .dropEffect wird automatisch auf "none" gesetzt "uninitialized" Default Standard-Dragging-Eigenschaft des Zieles verwenden

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
// ##### Quellobjekt #####
function EventHandlerFuerOnDragStart()
// Quellobjekt löst Event aus:
// in Quelle wird markiert und selektiert
{
// selektierte Quell-Daten in die Zwischenablage puffern
// (Puffer wird per window.event-Objekt verwaltet)
// Datentyp ist hier uninteressant, da für die Zwischenablage
// der Typ automatisch erkannt wird, also
// Speicherung der Daten in der Zwischenablage
// immer typgerecht ist
var ZwischenAblage = window.event.dataTransfer;

// und diese Daten als verschiebbar erklären:
// aus der Quelle in die Zwischenablage
ZwischenAblage.effectAllowed = "move";
}

// ##### Zielobjekt #####
function EventHandlerFuerOnDragEnter
// Zielobjekt löst Event aus
// Maus über Ziel nach dem Draggen der Daten,
// UND Maustaste ist noch nicht losgelassen
// also Zielobjekt wird mit den Daten betreten
{
// Daten adressieren
var ZwischenAblage = window.event.dataTransfer;

// und diese Daten als verschiebbar erklären:
// aus der Zwischenablage in das Zielobjekt
ZwischenAblage.dropEffect = "move";

// Event-Handler-Rückkehrcode
var EventObjekt = window.event;
EventObjekt.returnValue = false;
}

function EventHandlerFuerOnDrop
// Zielobjekt löst Event aus
// Maus über Ziel nach dem Droppen der Daten,
// UND Maustaste wird losgelassen
{
// Ziel adressieren, das mit ondrop-Ereignis behandelt werden soll
var Ziel = window.event.srcElement;
```



```

// Daten in der Zwischenablage adressieren und holen
//          (Puffer wird per window.event-Objekt verwaltet)
var ZwischenAblage = window.event.dataTransfer;

// und Daten vom Texttyp aus der Zwischenablage im Ziel ablegen,
//          da Ziel nur Textdaten empfangen kann
Ziel.innerText += Daten.getData("text");

// Event-Handler-Rückkehrcode
var EventObjekt = window.event;
EventObjekt.returnValue = false;
}

function EventHandlerFuerOnDragOver
// Zielobjekt löst Event aus
{
    // nichts tun ausser Rückkehrcode des Handlers liefern
    var EventObjekt = window.event;
    EventObjekt.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY>
    <DIV ondragstart=" EventHandlerFuerOnDragStart ()"
    >
        Diesen Text markieren und draggen
    </DIV>
    <BR>
    <DIV
        ondragover="EventHandlerFuerOnDragOver()"
        ondragenter="EventHandlerFuerOnDragEnter()"
        ondrop="EventHandlerFuerOnDrop()"
    >
        An diese Stelle den Text dropfen
    </DIV>
</BODY>
</HTML>

```

.enabled Sichtbarkeit des Media Bar Player
Media Bar Player ist der Windows Media Player
siehe Behavior .style.mediaBar

Beispiel:

```

<DIV     ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
>
    Div ist der Time-Container
</DIV>
<INPUT   TYPE=button
VALUE='abspielen von test.asx'
onclick= "ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
          ID_Div.disabledUI = true;
          ID_Div.enabled = false;
          "
>

```

.enctype Multipurpose Internet Mail Extensions (MIME) des Formulars für Encoding nach dem Senden
der Daten
ab IE 6.x

.end Objektaktivitäten laut Eigenschaft .timeAction beenden
ab IE 6.x
alternativ: Eigenschaft .dur
siehe Objekt currTimeState und Behavior .style.time2
Wert im Time-Format
z.B. "h:min:s.f"
 id.event
 id.event+zeit_wert_als_string

Beispiele für Kodierung von Time in der Eigenschaft .end:

```

object.end="objekt_zeiger.begin+10s"                   // warten bis Event "onbegin" zum Objekt laut
//                                                       objekt_zeiger (z.B. laut ID-Attribut)
//                                                       eintritt,
//                                                       dann 10 Sekunden warten
//                                                       dann Objekt laut object beenden

```



```

object.end="objekt_zeiger.focus+10s" // warten bis Event "onfocus" zum Objekt laut
//                                objekt_zeiger (z.B. laut ID-Attribut)
//                                eintritt,
//                                dann 10 Sekunden warten
//                                dann Objekt laut object beenden

object.end="2; objekt_zeiger.click+1" // 2 Sekunden nach dem Laden des Elternobjektes von
//                                object warten
//                                dann auf das Ereignis click zum Objekt laut
//                                objekt_zeiger warten
//                                dann 1 Sekunde warten
//                                dann Objekt laut object beenden

```

Hinweis: object laut ID-Attribut des Behavior-Objektes von .style.time2.

```

"25:45:10" 25 Stunden, 45 Minuten, 10 Sekunden
"45:35"    45 Minuten, 35 Sekunden
"45:00.275" 45 Minuten, 0,275 Sekunde
"10.5"     10,5 Sekunden

```

Beispiele für Kodierung von .end:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<SPAN ID="ID_Span1"
CLASS=time_line_klasse
STYLE="COLOR:Red;"
BEGIN="0"
END="10"
TIMEACTION="visibility"
>
<H3>Test 1</H3>
</SPAN>
<SPAN ID="ID_Span2"
CLASS=time_line_klasse
STYLE="COLOR:Blue;"
BEGIN="3"
END="10"
TIMEACTION="visibility"
>
<H3>Test 2</H3>
</SPAN>
<SPAN ID="ID_Span3"
CLASS=time_line_klasse
STYLE="COLOR:Green;"
BEGIN="6"
END="10"
TIMEACTION="visibility"
>
<H3>Test 3</H3>
</SPAN>
</BODY>
</HTML>

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter"
TYPE="fade"
DUR="8"
END="4"
TARGETELEMENT="ID_Div"

```




```

>
</ t:TRANSITIONFILTER>
<DIV ID="ID_Div"
      CLASS="time_line_klasse"
      DUR="indefinite"
      STYLE="background-image:url(test.gif); background-repeat: no-repeat;"
>
</DIV>
</BODY>
</HTML>

```

.endSync Ende der Animation von Elementen in einem gemeinsamen Time-Container
 bei Ende der Timeline des Eltern-Time-Containers
 Verhalten des Eltern-Time-Containers bezüglich der Animation seiner Kind-Elemente
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="parent"
        ENDSYNC="ID_Div2"
>
    <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="2"
    >
        Zeile 1
    </DIV>
    <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        BEGIN="2"
        DUR="2"
    >
        Zeile 2
    </DIV>
    <DIV ID="ID_Div3"
        CLASS="time_line_klasse"
        BEGIN="3"
        DUR="2"
    >
        Zeile 3
    </DIV>
</t:EXCL>
</BODY>
</HTML>

```

.event On-Eventbezeichner mit angefügten Klammernpaar
 Script soll das Event verwalten
 per script Objekt
 immer mit Eigenschaft `.htmlFor` kodieren

Beispiel:

```

<SCRIPT ID="ID_Script" FOR="ID_Botton" EVENT="onclick()">
var Text1 = "Pferdchen hue !"
var Text2 = "Pferdchen brrr !"

if (ID_Botton.innerText == Text1)
{ ID_Botton.innerText = Text2; }
else
{
    if (ID_Botton.innerText == Text2)
    { ID_Botton.innerText = Text1; }
}
</SCRIPT>
<BUTTON ID="ID_Botton" onmouseout="alert(ID_Script.event)">Hue oder Brrr ? </BUTTON>

```

.event Zeiger auf das Objekt event im Fenster
 siehe Objekt `window`



.expando Wirksamkeit von Attributen ein/aus
 true ein, Default
 false aus

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <SPAN ID="ID_Span" UNSELECTABLE="on">
        Dieser Text ist <B>selektierbar !!<B>
    </SPAN>
</DIV>
</BODY>
</HTML>
```

.expires Zeitstempel der Daten im UserDataStore (User-Cache) per .style.userData Behavior
 Zeitstempel als Verfallszeitpunkt für automatische Löschung gecachter Daten:
 Daten werden **automatisch** durch den Browser gelöscht, wenn das Datum der Daten anhand des
 Zeitstempels als verfallen erkannt wurde, also der aktuelle Zeitpunkt laut PC-Uhr jünger ist, als
 der Zeitstempel.
 String im UTC (Universal Time Coordinate)-Format

Beispiel:

```
<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //        es können diverse Cachenames definiert und somit Versionen von Cache
    //        verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //        es können diverse Attribute definiert und somit Versionen von Input-Daten
    //        verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache save
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
```



```

        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

.external Zeiger auf das Objekt window.external im Fenster

.face Familie des Font-Typs (Font-Face) z.B. Courier

Beispiel:

```

<FONT ID="ID_Font" FACE="Arial">
<SCRIPT>
    alert(ID_Font.face);
    ID_Font.face = 'Courier';
    alert(ID_Font.face);
</SCRIPT>

```

.fgColor Vordergrundfarbe (Textfarbe) im Dokument
#rrggbb Standard ist #000000
vordefinierter Farbname (browserspezifisch)

.fileCreatedDate Datum der Dokumenterstellung

Beispiel 1:

"Monday, December 08, 2000"

Beispiel 2:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

.fileModifiedDate Datum der letzten Dokumentveränderung

Beispiel:

"Monday, December 08, 2000"

.Files Zeiger auf die interne Collection FileSystemObject.Folder.Files
Collection kann nur z.T. über das JScript-Objekt Enumerator angesprochen werden

.fileSize Größe des Dokumentes

.FileSystem Typ des Dateisystems auf dem Laufwerk liefern

Beispiel:



```

var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk         = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Dateisystem = Laufwerk.FileSystem;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Dateisystem);

```

.fileUpdatedDate Datum des letzten Datei-Updates

Beispiel:

"Monday, December 08, 2000"

.fill

Aktion eines Elementes zum Zeitpunkt des Ende der Timeline des Elementes aber
bevor die Timeline des Elternelementes endet
ist Ersatz für die Eigenschaft endHold, die deprecated ist und nicht mehr verwendet
werden darf !!
siehe Objekt currTimeState und Behavior .style.time2

Beispiel 1:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="0"
DUR="15"
>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
BEGIN="0"
DUR="10"
FILL="freeze"
>
</DIV>
</t:EXCL>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<DIV STYLE="height:100px">
<t:SEQ ID="ID_Seq"
REPEATCOUNT="indefinite" >
<t:MEDIA ID="ID_Media1"
SRC = "test1.jpg"
STYLE="position:absolute;"
DUR="3"
TIMECONTAINER="par"
FILL="transition"
>
<t:TRANSITIONFILTER TYPE="fade"
DUR="2"
>
</t:TRANSITIONFILTER>
</t:MEDIA>
<t:MEDIA ID="ID_Media2"
SRC = "test2.jpg"
STYLE="position:absolute;"
DUR="3"
TIMECONTAINER="par"
FILL="transition"
>
<t:TRANSITIONFILTER ID="ID_Transfilter"

```



TYPE="ClockWipe"
DUR="2"

```

>
</t:TRANSITIONFILTER>
</t:MEDIA>
</t:SEQ>
</DIV>
</BODY>
</HTML>

```

.firstChild Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes

Beispiel:

```

<SCRIPT>
  var ZeigerAufErstesKind = Liste.firstChild; // liefert Referenz auf Listenelement 1
</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1
    <LI> Listenelement 2
    <LI> Listenelement 3
  </UL>
</BODY>

```

.fontSmoothingEnabled Fontglättung auf Bildschirm
per screen Objekt
false Default
keine Glättung
true Glättung ist aktiv

.form Zeiger auf das Formular (Formular als Container)
ab IE 6.x für Elemente fieldSet, label, legend

.frame Art des Rahmens um eine Tabelle
siehe Objekt table
"void" Standard, kein Rahmen
"above" Rahmen oberhalb
"below" Rahmen unterhalb
"border" Rahmen auf allen Seiten
"box" Rahmen auf allen Seiten
"hsides" Rahmen oben und unten
"lhs" Rahmen links
"rhs" Rahmen rechts
"vsides" Rahmen links und rechts

.frameBorder Borderanzeige ein/aus beim Frame
"0" oder "no" Anzeige aus
"1" oder "yes" Default
Anzeige ein
Hinweis: wenn Anzeige aus, so wird Eigenschaft .borderColor ignoriert

.frameElement Zeiger auf Frame bzw. IFrame, die im Fenster liegen
mit diesem Zeiger können im Fenster alle Eigenschaften und Methoden des Frame bzw. IFrame referenziert werden
siehe Objekt window

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
  var FrameZeiger = window.frameElement;
  FrameZeiger.src = "http://www.test.de ";
</SCRIPT>

```

.frames Zeiger auf die Collection document.frames im HTML-Dokument, das im Fenster angezeigt wird
siehe Objekt window

.frameSpacing Zwischenraum in Pixel zwischen 2 benachbarten Frames

.FreeSpace Freien Speicher in Bytes auf Laufwerk ermitteln

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_FreierPlatz = Laufwerk.FreeSpace;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_FreierPlatz);

```



.from Startwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline
 per .additive oder .accumulate
 für die Objekte animate, animateMotion und animateColor gilt:
 .from wird von .path und .values überschrieben
 Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function SpanInhaltInit(Kette)
{
    // Zustand
    ID_Span1.innerText = "";

    // Kumulationsart
    ID_Span2.innerText =Kette;

    // Zaehler auf 1
    ID_Span3.innerText ="1";

    // DIV-Text festlegen
    ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
    if (Kette == "sum")
    {
        // Wertkumulation von .width
        ID_Div.innerText += " genau 1x kumulativ um ";
        ID_Div.innerText += ID_Animate.by
        ID_Div.innerText += " mal "
        ID_Div.innerText += ID_Animate.repeatCount;
    }
    else
    {
        // keine Wertkumulation von .width
        ID_Div.innerText += ID_Animate.repeatCount;
        ID_Div.innerText += " mal um ";
        ID_Div.innerText += ID_Animate.by
    }

    ID_Div.innerText += " aus";
}

function KumulationAus()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("none");
    ID_Animate.accumulate="none";

    // DANACH Animation starten
    ID_Animate.beginElement();
}

function KumulationEin()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("sum");
    ID_Animate.accumulate="sum";

    // DANACH Animation starten
    ID_Animate.beginElement();
}

function Anzeige()
```



```

    {
        ID_Span1.innerText = "Animation ist beendet ";
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        FROM="150px"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

animierter DIV
<DIV ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE= "position:absolute;
        top:125px;
        left:25px;
        height:100px;
        width:125px;
        border:solid black 2px;
        "
    >
</DIV>
<BR>

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3"></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.from Start-Prozentsatz des kompletten Überganges bei Start der Animation des Übergangsfilters
 per `.style.time2.transitionFilter Behavior-Objekt`
 Start des Übergangsfilters mit bereits teilweise vollendetem Übergang
 nicht zusammen mit Eigenschaft `.value` kodieren, sonst wird `.from` ignoriert
 siehe Objekt `currTimeState` und Behavior `.style.time2`
 Ziffernfolge Floating point
 Wert numerisch von 0 bis 1
 Standard ist "0.0" oder "0"
 0 entspricht 0 % des kompletten Überganges
 1 entspricht 100% des kompletten Überganges
 Bsp.: 0.3 entspricht: Mit **bereits** 30% vollendeten kompletten Übergang die Animation starten

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:TRANSITIONFILTER ID="ID_Transfilter"
        FROM="0.3"
        BY="0.4"

```




```

        TYPE="barWipe"
        DUR="3"
        TARGETELEMENT="ID_Div"
    >
</t:TRANSITIONFILTER>

<DIV ID="ID_Div"
      CLASS="time_line_Klasse"
      DUR="indefinite"
      STYLE="position:relative; left:20px; width:420px; height:100px;
            background-image:url(test.gif); background-repeat: no-repeat;
            "
    >
</DIV>
</BODY>
</HTML>

```

.fromElement Referenz auf Maus-Eventauslösendes Quell-Element bei Mausbewegung
nicht für Event ondragleave
Objekt event

Beispiel:

```

<SCRIPT>
function TesteMaus(Zeiger)
{
    if( ! Zeiger.contains(event.fromElement) )
    { alert("Maus über Button erkannt"); }
}
</SCRIPT>
<BUTTON onmouseover=" TesteMaus(this)">Ueberfahre mich mit der Maus</BUTTON>

```

.galleryImg Toolbar "My Pictures Photo Support image toolbar" ein/ausschalten für aktuelles Image
true oder "yes" Default
Toolbar ein
false oder "no" Toolbar aus

Hinweis: bei Image-Mappe (USEMAP, ISMAP) wird Toolbar immer eingeschaltet
permanentes Einstellen der Toolbar für gesamte Webseite auch per META-Tag

HTTP-EQUIV="imagetoolbar" CONTENT="no" oder false bzw. "yes" oder true

Beispiel:

```

<IMG SRC="mypicture.jpg" HEIGHT="100px" WIDTH="100px" GALLERYIMG="yes">

```

.hasAudio Media-Datei mit Audioinhalt auf der Timeline
nicht alle Media-Typen können Audio beinhalten
Stummschaltung oder Lautstärkeinstellungen sind dabei egal
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function Anzeige()
{
    alert('hasAudio: ' + ID_Media.hasAudio);
}
</SCRIPT>
</HEAD>
<BODY>
<t:MEDIA ID="ID_Media"
        SRC="test.wmv"
        BEGIN="0"
        FILL="remove"
        onmediacomplete="Anzeige ();"
    >
</t:MEDIA>
</BODY>
</HTML>

```

.hasDownloadProgres Start des Donloades einer Media-Datei auf der Timeline
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:



```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                      SRC="test.avi"
                      SYNCBEHAVIOR="locked"
    >
    </t:VIDEO>
    <SPAN ID="ID_Span"
          CLASS="time_line_klasse"
          DUR="0.1"
          REPEATCOUNT="indefinite"
          onrepeat="ID_Span.innerText= ID_Video.hasDownloadProgress;"
    >
    </SPAN>
</BODY>
</HTML>

```

.hash Teil des Wertes der Eigenschaft **.href** also Teil der Url als Anker
 Teil folgt direkt hinter dem Nummernkreuz # und ohne Nummernkreuz
 lesen: ist Leerkette wenn kein # gefunden

Beispiel 1:

```
if (document.location.hash == "") { ... }
```

.hasLayout Objekt mit Style-Layout
 ab IE 5.5
 Style-Layout wird erzeugt durch Kodierung einer Style-Eigenschaft (siehe style Objekt)
 oder durch Setzen der Eigenschaft **.contentEditable** auf true
 Folgende Objekte haben immer ein Style-Layout:
 BODY, IMG, INPUT, TABLE, TD

Objekte mit	.style.display	auf inline-block
	.style.height	
	float	auf left oder right
	.style.position	auf absolute
	.style.width	
	.style.writingMode	auf tb-rl
	.style.zoom	

false Default.
 Objekt hat kein Layout
 true Objekt hat Layout

.hasMedia Objekt ist HTML-Media-Objekt
 siehe Objekt **currentTimeState** und Behavior **.style.time2**

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    { alert('hasMedia = ' + ID_Media.hasMedia); }
</SCRIPT>
</HEAD>
<BODY>
    <t:MEDIA          ID="ID_Media"
                      BEGIN="0"
                      SRC="/test /media/test.wmv"
                      FILL="remove"
                      onmediacomplete="Anzeige();"
    >
    </ t:MEDIA>
</BODY>
</HTML>

```

.hasNextItem aktueller Eintrag in der Playliste (Objekt **PlaylistInfo**) hat einen Nachfolger-Eintrag
 siehe Behavior **.style.mediaBar**
 false Default



kein Nachfolger
true mit Nachfolger

.hasPlayList Verfügbarkeit der Behavior-Collection `.style.time2.playList` für Element auf der Timeline, also ob Element eine Playliste hat
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    { alert('hasPlayList: ' + ID_Media.hasPlayList); }
</SCRIPT>
</HEAD>
<BODY>
    <t:MEDIA          ID="ID_Media"
                      SRC="test.wmv"
                      BEGIN="0"
                      FILL="remove"
                      onmediacomplete="Anzeige();"
    >
    </t:MEDIA>
</BODY>
</HTML>
```

.hasVisual Media-Datei mit visuellem Inhalt auf der Timeline
visuelle Daten werden auf dem Bildschirm sichtbar
nicht alle Media-Typen können sichtbare Daten beinhalten
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        alert('hasAudio: ' + ID_Media.hasVisual);
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:MEDIA          ID="ID_Media"
                      SRC="test.wmv"
                      BEGIN="0"
                      FILL="remove"
                      onmediacomplete="Anzeige ();"
    >
    </t:MEDIA>
</BODY>
</HTML>
```

.height Höhe des Objektes in Pixel

.height Auflösung des Bildschirms in Höhe, also Anzahl der vertikalen Pixel
per screen Objekt

.height Auflösung des Bildschirms in Höhe, also Anzahl der vertikalen Pixel
Behavior `.style.clientCaps`

.hideFocus Focussierbarkeit
true Focus ist nicht möglich
false Default
Focus ist möglich

Beispiel:

```
<BUTTON>fokussierbar</BUTTON>
<BUTTON HIDEFOCUS="true">nicht fokussierbar</BUTTON>
```



.higher Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines höher priorisierten Kindes
 Time-Container ist Behavior-Objektes `.style.time2.excl`
 Folge der in HTML kodierten Objekte `t:PRIORITYCLASS` entspricht der absteigenden Folge der Prioritäten der Animation.
 Jedes Kind ist ein `t:PRIORITYCLASS`
 darf **kein** weiteres Behavior-Objekt `.style.time2.priorityClass` besitzen
 wenn mehrere pausierende Kinder existieren:
 Das zuletzt pausierte Kind startet zuerst.
 Das zuerst pausierte Kind startet zuletzt.
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
  BEGIN="0;indefinite;"
>
  <t:PRIORITYCLASS>
    <SPAN ID="ID_Span1"
      CLASS="time_line_Klasse"
      BEGIN="5"
      DUR="5"
    >
      Test1
    </SPAN>
  </t:PRIORITYCLASS>
  <t:PRIORITYCLASS HIGHER="stop">
    <SPAN ID="ID_Span2"
      CLASS="time_line_Klasse"
      BEGIN="0"
      DUR="10"
    >
      Test2
    </SPAN>
  </t:PRIORITYCLASS>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
</BUTTON>
</BODY>
</HTML>
```

.history Zeiger auf das Objekt history (Verlauf)
 siehe Objekt window

.host **Hostname** und **Port** einer Location oder Url in der Form "**hostname:port**"

Beispiel:

```
document.location.host;
```

.hostname **Hostname** einer Location oder Url in der Form "**hostname:port**"
 kann Domain oder IP sein

.href Ziel-Url oder Anker als Sprungziel (z.B. `<A>`-Tag)
 siehe auch Eigenschaften `.rel` und `.rev`

Beispiel 1:

```
document.location.href;
```

Beispiel 2 für globalen Style per HEAD:

```
<HEAD>
<STYLE>
.an {text-decoration: underline overline; color:blue;}
.aus {text-decoration: none; color:black;}
</STYLE>
</HEAD>
<BODY>
<A      HREF="test.htm"
```



```

        CLASS="aus"
        onmouseover="this.className='an';"
        onmouseout="this.className='aus';"
    >
    </A>
</BODY>

```

Beispiel 3 für Abschaltung Rahmen um Link:

Immer, wenn Sie auf einen Link oder einen Button klicken, zieht der Internet Explorer einen kleinen, gestrichelten Rahmen darum, der erst beim Klick auf ein anderes Objekt wieder verschwindet. Dieser Rahmen ist vor allem bei Imagemaps störend. Der HTML-Code für das Objekt (z.B. einen Link) ist onclick="this.blur()" zu erweitern.

```
<a href="seite.htm" onclick="this.blur()">Link</a>
```

Beispiel 4 für Webseite mit eigenem Icon ab IE 5.x:

```
<LINK REL="SHORTCUT ICON" HREF="name.ico">
```

name.ico: name ist beliebig, auch mit Pfad

ICO-Datei: 32x32 Pixel mit 16 Farben
oder 16x16 Pixel mit 256 Farben

ist BMP-Datei, die zu ICO-Datei konvertiert werden muss

(kann nicht jedes Grafik-Programm, aber Microsoft bietet dafür IconPro an)

.href Url einer externen Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
siehe Objekt styleSheet

.hreflang Sprachcode des Objektes laut RFC1766

Beispiel:

```
<A HREF="www.test.com" HREFLANG="en-US">Link</A>
```

.hspace horizontaler Abstand in Pixel zum Elternobjekt

.htmlFor ID des Label **nur für die Realisierung der Beschriftung**
ist nicht das ID laut ID-Attribut
per label Objekt

.htmlFor Referenz auf Objekt, das das Event laut Eigenschaft .event verwalten soll
und dafür einen Scriptaufruf im Ereignishandler onxxx besitzt
per script Objekt
immer mit Eigenschaft .event kodieren

Beispiel:

```

<SCRIPT ID="ID_Script" FOR="ID_Botton" EVENT="onclick()">
    var Text1 = "Pferdchen hue !"
    var Text2 = "Pferdchen brrr !"

    if (ID_Botton.innerText == Text1)
    { ID_Botton.innerText = Text2; }
    else
    {
        if (ID_Botton.innerText == Text2)
        { ID_Botton.innerText = Text1; }
    }
</SCRIPT>
<BUTTON ID="ID_Botton" onmouseout="alert(ID_Script.event)">Hue oder Brrr ? </BUTTON>

```

.htmlText HTML-Text im Textbereich
nicht den Plaintext-Anteil liefern
per textrange Objekt
nur unter Windows 32-Bit

.httpEquiv Informationen des HTTP Response Header per meta Objekt
Beschaffung der Informationen per Serverfunktionen
HttpQueryInfo und QueryInfo
Dieser Kopf wird von Suchmaschine, Server und Browser verwendet
der Wert der Information wird in der Eigenschaft .content abgelegt

Beispiele:

```

Dokument alle 2 Sekunden neu laden
<META HTTP-EQUIV="REFRESH" CONTENT=2>

```

```

Zeichensatz des Dokumentes
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=Windows-1251">

```



Themenunterstützung abschalten

```
<META HTTP-EQUIV="MSTHEMECOMPATIBLE" CONTENT="no">
```

HTML-Seiten-Refresh für WebCam

```
<META HTTP-EQUIV="refresh" CONTENT="sekundenzahl">
```

```
<META HTTP-EQUIV="Expires" CONTENT="Tue, 04 Dec 1997 21:29:026 GMT">
```

Datum egal, aber auf jedenfall älter als aktuelles, damit die HTML-Seite als verfallen gilt

```
<META HTTP-EQUIV="Pragma" CONTENT="nocache">
```

Seite nicht in Browsercache speichern

```
<META HTTP-EQUIV="Cache-Control" CONTENT="nocache">
```

Seite nicht aus Browsercache laden

um sicher zu gehen, daß immer die Seite mit garantiert aktuellem Stand geladen wird

→ eventuell ist Proxyserver nicht aktuell

```
<META HTTP-EQUIV="expires" CONTENT=sekunden_wert>
```

sekunden_wert: Wartezeit in Sekunden bis zum nächsten Laden unter Umgehung des Proxyserver-Cache → 0, also immer umgehen und sofort laden

.id

Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname)

ID-Attribut in HTML: Wert ist String

alphanumerisch

muss mit Buchstaben beginnen

Unterstrich _ verwendbar

kann in " " bzw. ' ' kodiert werden, muss aber nicht

Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen).

Zeiger aus ID bilden var Zeiger = eval(object.id);

Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.

Beispiel:

```
<SCRIPT>
```

```
function Anzeige(ObjektZeiger)
```

```
{alert('ID = ' + ObjektZeiger.id);}
```

```
</SCRIPT>
```

```
<TABLE BORDER COLS=3>
```

```
<TR>
```

```
<TD ID="ID_Zelle1" onclick=" Anzeige(this);">Zelle 1</TD>
```

```
<TD ID="ID_Zelle2" onclick=" Anzeige(this);">Zelle 2</TD>
```

```
<TD ID="ID_Zelle3" onclick=" Anzeige(this);">Zelle 3</TD>
```

```
</TR>
```

```
</TABLE>
```

IMMEDIATEEND

wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software

.implementation

Referenz auf Objekt implementation zum Dokument

.indeterminate

Grauzustand (Dimmed) **und** Selektiertheit des Checkbox-Control

Achtung: verändert Eigenschaften .checked .status und .defaultChecked

verändert **nicht** Anwählbarkeit durch User (Attribut DISABLED, Eigenschaft .disabled) also ist der Focus weiterhin veränderbar !

wenn User auf per Eigenschaft .indeterminate angegrautetes

Checkbox-Control klickt, so wird

Angegrautheit aufgehoben (.indeterminate auf false)

und der Status geändert (.status etc. neu gesetzt)

false Default

nicht selektiert **und** nicht grau

true selektiert **und** grau

.index

laufende Nummer der Option, also Objektes document.select.option, in einer List-Box als document.select Objekt

.index

Index des Objektes playItem in der Behavior-Collection .style.time2.playList

Track entspricht playItem Objekt

Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:

Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline

aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.

Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```
var Wert = object.playList.index; // Wert Integer, ab 0
```

```
var Wert = object.playList.aktiveTrack.index; // Wert Integer, ab 0
```



Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        alert('Index: ' + ID_Media.playlist.activeTrack.index);
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:MEDIA          ID="ID_Media"
                      SCR="test.asx"
                      BEGIN="0;"
    >
    </t:MEDIA>
    <BUTTON          ID="ID_Button"
                    onclick="Anzeige();"
    >
    </BUTTON>
</BODY>
</HTML>
```

.index
Index des Playlisten-Eintrages in der Behavior-Collection .style.time2.playlist
Diese Eigenschaft ist eigentlich nur sinnig für die Kodierung zum aktiven Track..
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
var PlaylistElement = zeiger_auf_timer_objekt.playlist.item(Wert);
// Wert      Index in der Behavior-Collection .style.time2.playlist
//              ab 0
```

.....

```
var VergessenerIndex = PlaylistElement.index;

var IndexAktiverTrack = zeiger_auf_timer_objekt.playlist.activeTrack.index;

// zeiger_auf_timer_objekt      laut ID-Attribut
```

.innerHTML
Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
also nach dem Laden des Objektes
aber wirksam erst mit parsen des HTML-Endetags
Plain-Text
HTML-Elemente je nach Objekt möglich
Script: muss mit DEFER-Attribut kodiert sein
schreiben: wenn Bereich nicht leer, so komplett überschreiben
wenn Bereich leer so einfügen
nur lesen bei COL, COLGROUP, FRAMESET, HTML, **STYLE**, TABLE, TBODY, TFOOT, THEAD,
TITLE, TR.

Beispiel 1:

```
<P onmouseover="this.innerHTML='<B>on Mouse Over erkannt</B>'"
  onmouseout="this.innerHTML='<I>on Mouse Out erkannt</I>'">
  <I>Dieser Text wird veraendert</I>
</P>
```

Beispiel 2:

```
<HTML>
<SCRIPT>
    function Veraendern()
    {
        // Scriptcode erzeugen zur Laufzeit
        var ScriptText = "<SCRIPT DEFER>"; // muss DEFER sein !!!
        ScriptText = ScriptText
            + "function Test(){ alert('Hallo von Script.')}";
        ScriptText = ScriptText
            + "</SCRIPT>" + ">";

        // neuer Text erzeugen zur Laufzeit
```




```

var TextMitHTML = "<input type=button onclick="
+ "Test()" // Aufruf der Funktion im Script
+ " value='Click Me'><BR>";

// Text und Script als neuer Inhalt des Div
ID_Div.innerHTML = TextMitHTML + ScriptText;
}
</SCRIPT>
<BODY onload="Veraendern();">
  <DIV ID="ID_Div">Alter Inhalt</DIV>
</BODY>
</HTML>

```

Beispiel 3: Anstelle der Alert-Box wird oft ein per STYLE-Attribut wohl positionierter DIV benutzt

```

<DIV ID="ID_Div" STYLE="....">Hier erscheint der Kommentar</DIV>
in dem ein Text mit oder ohne HTML-Tags angezeigt wird. Dieser DIV wird in seiner Eigenschaft
ID_Div.innerHTML durch ein anderes HTML-Element gefüllt, das einen Kommentar loswerden will.
ID_Div.innerHTML="<B>Das ist ein fetter Kommentar</B>";

```

Hinweis: Die Status-Zeile des Browserfenster sollte **nur den Status-Meldungen** des Browsers vorbehalten sein.

Beispiel 4 für Ersatz von document.write() durch die Eigenschaft .innerHTML (nur IE):

```

<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette; // wird sofort geparkt, also IMG-Tag ausgeführt
                          // (anstelle von eval()
                          // bzw. document.write() )

    ID_IMG.SRC="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>

```

.innerText Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag
 ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B.
 dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,
 also nach dem Laden des Objektes
 aber wirksam erst mit parsen des HTML-Endetags
 nur Plain-Text also keine HTML-Elemente und kein Script
 schreiben: wenn Bereich nicht leer, so komplett überschreiben
 wenn Bereich leer so einfügen
 nur lesen bei HTML, TABLE, TBODY, TFOOT, THEAD, TR

Beispiel 1:

```

<P ID="ID_P">Dieser Text wird verändert</P>
<BUTTON onclick=" ID_P.innerText='Neuer Text'">Neuer Text</BUTTON>
<BUTTON onclick=" ID_P.innerText= Dieser Text wird verändert">Reset</BUTTON>

```

Beispiel 2: Anstelle der Alert-Box wird oft ein per STYLE-Attribut wohl positionierter DIV benutzt

```

<DIV ID="ID_Div" STYLE="....">Hier erscheint der Kommentar</DIV>
in dem ein Text ohne HTML-Tags angezeigt wird. Dieser DIV wird in seiner Eigenschaft
ID_Div.innerText durch ein anderes HTML-Element gefüllt, das einen Kommentar loswerden will.
ID_Div.innerText="Das ist ein Kommentar";

```

Hinweis: Die Status-Zeile des Browserfenster sollte **nur den Status-Meldungen** des Browsers vorbehalten sein.

Beispiel 3 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
// ++++++ globale Variablen, die verändert werden können
var SoundUrl = "56sec.mid";

```



```

var SoundDauerInSekunden    = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++ Routinen der Sekundenzählung
var SekundenZahler          = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen()   // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekundenzählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekundenzählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl          = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
                                                    // Timerzeit für Rekursion
    this.SoundFileBeendet      = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen

```



```

        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString()+ " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdrucks.
    //      Für die Neuberechnung des Ausdrucks ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
        "SekundenZahler * PixelBreiteProBalkenErweiterung"
    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText = "Der Sound dauert "
        + SoundDauerInSekunden.toString()
        + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
        + 'STYLE="background-color:lightblue"'
        + '>'
        + '</DIV>'

```



```

        + '<BR>'
    );

    document.write(
        '<DIV ID="ID_DIV_SekundenZahler"'
        + 'STYLE="color:hotpink;font-weight:bold"'
        + '>'
        + '</DIV>'
        + '<BR>'
    );

    document.write(
        '<DIV ID="ID_DIV_MessLatte"'
        + 'STYLE="color:white;background-color:gray"'
        + '>'
        + '</DIV>'
    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ----- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

.isContentEditable Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
 Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
 false Content nicht editierbar
 true Content editierbar

Beispiel:

```

<HEAD>
<SCRIPT>
    function EditierbarkeitWechseln()
    {
        var Editierbarkeit = ID_Span1.isContentEditable;
        var Editierbarkeit_Negation = ! Editierbarkeit;
        ID_Span1.contentEditable = Editierbarkeit_Negation;
        ID_Span2.innerText = Editierbarkeit_Negation;

        if (Editierbarkeit_Negation == false)
        { ID_Button.innerText="editierbar machen" }
        else
        { ID_Button.innerText="nicht editierbar machen" }
    }
</SCRIPT>
</HEAD>
<BODY onload=" ID_Span2.innerText = ID_Span1.isContentEditable;">
    <BUTTON ID="ID_Button" onclick=" EditierbarkeitWechseln()">
        Klick mich
    </BUTTON>
    <SPAN ID="ID_Span1">Diesen Text editieren</SPAN>
    <SPAN ID="ID_Span2">Editierbarkeit</SPAN>
</BODY>

```

.isDisabled Interaktionsfähigkeit
 nur wenn sichtbar, so User-Interaktion möglich
 false User kann mit Objekt interagieren
 true User kann mit Objekt nicht interagieren

Beispiel:

```

<HEAD>
<SCRIPT>
    function SichtbarkeitWechseln()
    {
        var Sichtbarkeit = ID_Span1.isDisabled;
        var Sichtbarkeit_Negation = ! Sichtbarkeit;
        ID_Span1.isDisabled = Sichtbarkeit_Negation;
        ID_Span2.innerText = Sichtbarkeit_Negation;
    }

```



```

        if (Sichtbarkeit_Negation == false)
        { ID_Button.innerText="interaktiv machen"}
        else
        { ID_Button.innerText="nicht interaktiv machen"}
    }
</SCRIPT>
</HEAD>
<BODY onload=" ID_Span2.innerText = ID_Span1.isDisabled;">
    <BUTTON ID="ID_Button" onclick="SichtbarkeitWechseln()">
        Klick mich
    </BUTTON>
    <SPAN ID="ID_Span1">Diesen Text editieren</SPAN>
    <SPAN ID="ID_Span2">Sichtbarkeit</SPAN>
</BODY>

```

.isMap server-seitige Image-Map

.isMultiLine Mehrzeiligkeit des Objekthinhaltes
 false Objekt hat genau 1 Zeile
 true Objekt hat mindestens 1 Zeile

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Antworten(Wert)
    {Wert ? alert("Ja") : alert("Nein");}
</SCRIPT>
</HEAD>
<BODY>
    <P>
        <INPUT type="text" ID="ID_Input" VALUE="Test">
        <BUTTON onclick="Antworten(ID_Input.isMultiLine);">
            Mehrzeiligkeit eines INPUT TYPE=text
        </BUTTON>
    </P>
    <P>
        TEXTAREA:
        <TEXTAREA ID="ID_Textarea">
            Test
        </TEXTAREA>
        <BUTTON onclick="Antworten(ID_Textarea.isMultiLine);">
            Mehrzeiligkeit eines Textbereiches
        </BUTTON>
    </P>
</BODY>
</HTML>

```

.isMuted Elemente-Status der Audio-Stummschaltung in der Timeline
 siehe Objekt currTimeState und Behavior .style.time2
 true Audio ist stummgeschaltet
 false Audio ist hörbar

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Stummschaltung()
    {
        if (ID_Video.currTimeState.isActive)
        { ID_Span.innerText=' stumm = ' + ID_Video.currTimeState.isMuted;}
        else
        { ID_Span.innerText=' stumm =';}
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:media ID="ID_Video"
        SRC="/test /media/test.wmv"
        BEGIN="indefinite;"
        FILL="remove"
    >

```



[illegible]

.isOn

Elemente-Status aktiv oder halten in der Timeline
 Stauts aktiv: Objekt kann auf Events reagieren
 Status halten: Attribut Fill muss auf "freeze" oder "hold" gesetzt sein
 ermittelbar per eigenschaft .state
 Objekt kann nicht auf Events reagieren
 zu Attribut FILL:
 wenn "hold" oder "freeze" dann wartet Element auf
 die Synchronisation mit der
 Timeline des Elternobjektes
 wenn "hold" Element kann keine Events verarbeiten
 und ist inaktiv
 wenn "freeze" Element ist aktiv

siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function WindowOnLoad()
    {window.setInterval(UpdateZeit, 10);}

    function UpdateZeit()
    {
        if (ID_Animation.currTimeState.isOn == true )
        { ID_Span.innerHTML="aktiv";}
        else
        { ID_Span.innerHTML="gehalten";}
    }

    window.onload = WindowOnLoad;
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(ID_Animation.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <t:EXCL ID="ID_excl"
        BEGIN="indefinite"
        DUR="5"
        REPEATCOUNT="5"
    >

```



```

        <t:ANIMATEMOTION
            ID="ID_Animation"
            TARGETELEMENT="ID_Div"
            TO="200,0"
            BEGIN="0"
            DUR="2"
            AUTOREVERSE="true"
        >
    </t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
    CLASS="time"
    STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<SPAN ID="ID_Span">
    not
</SPAN>
<BUTTON onclick="ID_excl.beginElement();">Start</BUTTON>
<BUTTON onclick="ID_excl.endElement();">Stop</BUTTON>
</BODY>
</HTML>

```

.isOpen

prüfen ob ein per .createPopup() instanziiertes Popup-Fenster angezeigt wird
 siehe Objekt window.popup
 true, so angezeigt
 false, so nicht angezeigt

Beispiel für diverse Meldungsfenster per **jeweiligem** PopUp-Fenster (es kann immer nur genau 1 Popup-Fenster angezeigt werden):

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">

    var PopUpFensterFeld = new Array();

    // nachfolgende Felder beschreiben die PopUp-Fenster und müssen
    // identische Anzahl der Feldelemente haben
    // Feld-Index ist die Nummer des PopUp-Fensters

    var PopUpFenster_RahmenStyle_Feld = new Array
    (
        "solid black 4px",
        "solid blue 3px",
        "solid green 2px"
    );

    var PopUpFenster_HintergrundFarbe_Feld = new Array
    (
        "yellow",
        "gray",
        "orange"
    );

    var PopUpFenster_Inhalt_Feld = new Array
    (
        "<B>Inhalt PopUpFenster 0<B>",
        "Inhalt PopUpFenster 1",
        "<TT>Inhalt PopUpFenster 2<TT>"
    );

    var PopUpFenster_X_Koordinate_Feld = new Array
    (
        100,
        150,
        200
    );

    var PopUpFenster_Y_Koordinate_Feld = new Array
    (
        150,
        200,
        250
    );

```




```

var PopUpFenster_Breite_Koordinate_Feld = new Array
(
    80,
    140,
    200
);

var PopUpFenster_Hoehe_Koordinate_Feld = new Array
(
    100,
    140,
    180
);

var AnzahlPopUpFenster = PopUpFenster_Hoehe_Koordinate_Feld.length;

function PopupFensterInstanzieren(NummerDesPopUpFensters, ZeigerAufElternFenster)
    // NummerDesPopUpFensters ab 0
{
    PopUpFensterFeld[NummerDesPopUpFensters] =
        ZeigerAufElternFenster.createPopup();
}

function PopupFensterFuellen(NummerDesPopUpFensters)
{
    // Body des Popup-Fensters gestalten
    var PopupFenster_Body =
        PopUpFensterFeld[NummerDesPopUpFensters].document.body;

    PopupFenster_Body.style.backgroundColor =
        PopUpFenster_HintergrundFarbe_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.style.border =
        PopUpFenster_RahmenStyle_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.innerHTML =
        PopUpFenster_Inhalt_Feld[NummerDesPopUpFensters];
}

function PopupFensterAnzeigen(NummerDesPopUpFensters,
    ObjektZuDemPopUpFensterRelativPositioniertIst
)
{
    // Popup-Fenster anzeigen und damit öffnen
    PopUpFensterFeld[NummerDesPopUpFensters].show(
        PopUpFenster_X_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Y_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Breite_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Hoehe_Koordinate_Feld[NummerDesPopUpFensters],
        ObjektZuDemPopUpFensterRelativPositioniertIst
    );
}

function PopupFensterSchliessen(NummerDesPopUpFensters)
{
    var Zeiger = PopUpFensterFeld[NummerDesPopUpFensters];

    if (Zeiger.isOpen)
    { Zeiger.hide(); }
}

function Init()
{
    // PopUpFenster instanzieren im aktuellen Fenster (window)
    for (var i = 0 ; i < AnzahlPopUpFenster; i++)
    {
        PopupFensterInstanzieren(i, window);
        PopupFensterFuellen(i);
    }
}
</SCRIPT>
</HEAD>

```



```

<BODY onload="Init();">
    Durch Klick ausserhalb des Popup-Fensters wird dieses auch automatisch geschlossen.
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 anzeigen"
        onclick=" PopupFensterAnzeigen(0, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 anzeigen"
        onclick=" PopupFensterAnzeigen(1, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 anzeigen"
        onclick=" PopupFensterAnzeigen(2, document.body);"
    >
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 schliessen"
        onclick=" PopupFensterSchliessen(0);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 schliessen"
        onclick=" PopupFensterSchliessen(1);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 schliessen"
        onclick=" PopupFensterSchliessen(2);"
    >
</BODY>
</HTML>

```

.isPaused

Elemente-Status pausieren in der Timeline
 siehe Objekt currTimeState und Behavior .style.time2

true	Element pausiert wartet auf Aktivierung kann keine anderen Events verarbeiten
false	Default Element pausiert nicht und ist aktiv

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Pausieren()
    {
        if (ID_Animation.currTimeState.isPaused == true)
        { ID_Span.innerHTML="pausiert"; }
        else
        { ID_Span.innerHTML="aktiv"; }
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID=ID_Span"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(ID_Animation.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <t:EXCL ID="ID_excl"
        BEGIN="indefinite"
        DUR="5"
        REPEATCOUNT="5"
        onbegin="Pausieren();"
        onend=" ID_Span.innerHTML='pausiert';
    >
        <t:ANIMATEMOTION ID="ID_Animation"
            TARGETELEMENT="ID_Div"
            TO="200,0"

```



```

        BEGIN="0"
        DUR="2"
        AUTOREVERSE="true"
    >
    </t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
    CLASS="time"
    STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<SPAN ID="ID_Span">not</SPAN>
<BUTTON onclick="ID_excl.beginElement();">
    Start
</BUTTON>
<BUTTON onclick="ID_excl.pauseElement();Pausieren();">
    Pause
</BUTTON>
<BUTTON onclick="ID_excl.resumeElement();Pausieren();">
    Resume
</BUTTON>
<BUTTON onclick="ID_excl.endElement();ID_Span.innerHTML='pausiert';">
    Stop
</BUTTON>
</BODY>
</HTML>

```

.IsReady Bereitschaft des Laufwerkes für Zugriffe z.B. ob Medium im Laufwerk liegt

Beispiel:

```

var DateiSystem                = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung      = DateiSystem.GetDriveName("C:");
var Laufwerk                   = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName       = Laufwerk.VolumeName;
var Laufwerk_Bereitschaft      = Laufwerk.IsReady;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Bereitschaft);

```

.IsRootFolder prüfen auf Root

Beispiel:

```

var OrdnerName                = "c:\\windows\\desktop\\";
var DateiSystem               = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                    = DateiSystem.GetFolder(OrdnerName);
var Zahler                    = 0;
if (!Ordner.IsRootFolder)
{
    do
    {
        Ordner = Ordner.ParentFolder;
        Zahler++;
    }
    while (!Ordner.IsRootFolder);
}

alert("Ordner liegt " + Zahler + " Ebenen unter der Root");

```

.isStreamed Media-Datei mit Datenfluss (Data Stream) auf der Timeline
nicht alle Media-Typen unterstützen Datenstrom
siehe Objekt currTimeState und Behavior .style.time2
false hat keinen Datenfluss
true hat Datenfluss

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        SYNCBEHAVIOR="locked"
    >
    </t:VIDEO>
    <SPAN ID="ID_Span"

```



```

        CLASS="time_line_klasse"
        DUR="0.1"
        REPEATCOUNT="indefinite"
        onrepeat="ID_Span.innerText= ID_Video.isStreamed;"
    >
</SPAN>
</BODY>
</HTML>

```

.isTextEdit Erzeugbarkeit eines Textbereiches
 z.B. bei `BUTTON`
`TEXTAREA`
`INPUT BUTTON`
`INPUT HIDDEN`
`INPUT PASSWORD`
`INPUT RESET`
`INPUT SUBMIT`
`INPUT TEXT`
 false Textbereich nicht erzeugbar
 true Textbereich erzeugbar

.javaEnabled Verfügbarkeit der Microsoft Virtual Machine (Microsoft VM) für Java
 Achtung: Aufgrund eines Rechtsstreites mit dem Unternehmen Sun entwickelt Microsoft ab IE 6.x die MS VM nicht mehr weiter, hält die VM aber noch per Download verfügbar. Für eine aktuellere Version muss der User selbständig beim Unternehmen Sun eine Lizenz für Java erwerben und die zugehörige Software installieren (inklusive Updates). Es reicht dabei die Run-Time-Version der Virtuellen Java Maschine von Sun. Ob allerdings Sun-Java-Standards in Microsoft-Produkten implementiert werden, ist nicht gesichert.
 Behavior `.style.clientCaps`
 false Microsoft VM ist nicht verfügbar
 true Microsoft VM ist verfügbar

.keyCode Unicode der Taste, die das Event auslöst per `onkeydown`, `onkeyup` und `onkeypress`
 Objekt event

.keySplines Wert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline
 per `.additive` oder `.accumulate`
 benötigt `.calcMode` auf "spline"
`.keyTimes`
`.values` oder `.path`
 Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel 1:

KEYSPLINES="0 1 .5 1;5 1 0 1"

Quadrupel	Koordinaten			
	x1	y1	x2	y2
1	0	1	.5	1
2	5	1	0	1

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
.div_Klasse{ position:absolute; top:195px; height:100px; width:150px; border:solid black 2px;}
</STYLE>
</HEAD>
<BODY>
<SPAN ID="ID_Span1"
CLASS="time_line_klasse"
DUR=".1"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=( ID_Animatemotion.currTimeState.simpleTime);"
>
0
</SPAN>
<BR>
<SPAN ID=" ID_Span2"
CLASS="time_line_klasse"
DUR=".1"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=( ID_Animatemotion.currTimeState.activeTime);"

```



```

>
0
</SPAN>
<DIV ID="ID_Div"
CLASSE="div_klasse"
>
animierter Div
</DIV>
<t:ANIMATEMOTION ID="ID_Animatemotion"
BEGIN="1; ID_Button.click+1"
DUR="6s"
AUTOREVERSE="true"
CALCMODE="spline"
KEYSPLINES="0 1 .5 1; .5 1 0 1"
KEYTIMES="0;.5;1"
VALUES="25,0;250,50;500,0"
TARGETELEMENT="ID_Div"
FILL="hold"
>
</t:ANIMATEMOTION>
<BUTTON ID="ID_Button">Restart</BUTTON>
</BODY>
</HTML>

```

.keyTimes Zeitwert eines Pixel-Intervalles zur Werterhöhung bei 2D-Elemente-Animation(en) auf der Timeline
 per .additive oder .accumulate
 benötigt .calcMode auf "spline"
 .keySplines
 .values oder .path
 Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
.div_Klasse{position:absolute; top:195px; height:100px; width:150px; border:solid black 2px;}
</STYLE>
</HEAD>
<BODY>
<SPAN ID="ID_Span1"
CLASS="time_line_klasse"
DUR=".1"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=( ID_Animatemotion.currTimeState.simpleTime);"
>
0
</SPAN>
<BR>
<SPAN ID="ID_Span2"
CLASS="time_line_klasse"
DUR=".1"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=( ID_Animatemotion.currTimeState.activeTime);"
>
0
</SPAN>
<DIV ID="ID_Div"
CLASSE="div_klasse"
>
animierter Div
</DIV>
<t:ANIMATEMOTION ID="ID_Animatemotion"
BEGIN="1; ID_Button.click+1"
DUR="6s"
AUTOREVERSE="true"
CALCMODE="spline"
KEYSPLINES="0 1 .5 1; .5 1 0 1"
KEYTIMES="0;.5;1"
VALUES="25,0;250,50;500,0"
TARGETELEMENT="ID_Div"

```



```

        >
        </t:ANIMATEMOTION>
        <BUTTON ID="ID_Button">Restart</BUTTON>
    </BODY>
</HTML>

```

.label Label einer Option, also Objektes document.select.option des Objektes document.select

.lang Sprache für Anzeige von Sonderzeichen etc.

.language Sprache für Script festlegen
 gross klein egal
 "JScript" (Default)
 "javascript"
 "vbs" oder "vbscript"
 "XML" ab IE 5.x

.lastChild Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes

Beispiel:

```

<SCRIPT>
    var ZeigerAufLetztesKind = Liste.lastChild; // liefert Referenz auf Listenelement 3
</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listenelement 2
        <LI> Listenelement 3
    </UL>
</BODY>

```

.lastModified Datum der letzten Dokumentveränderung

.latestMediaTime Wartezeit für Start der Wiedergabe eines Media-Elementes auf der Timeline
 nicht für Media-Datei mit Datenfluss
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        SYNCBEHAVIOR="locked"
    >
    </t:VIDEO>
    <SPAN ID="ID_Span"
        CLASS="time_line_klasse"
        DUR="0.1"
        REPEATCOUNT="indefinite"
        onrepeat="ID_Span.innerText= ID_Video.latestMediaTime;"
    >
    </SPAN>
</BODY>
</HTML>

```

.left linke Pixelposition des Rechteckes um ein Objekt
 auch textrectangle Objekt

Beispiel für Nicht-TextRectangle-Objekt:

```

<SCRIPT>
    function Anzeigen(ObjektZeiger)
    {
        var Rechteck = ObjektZeiger.getBoudingClientRect();

        alert(
            "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
            + "\n"
            + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
        );
    }

```



```
</SCRIPT>
<P onclick="Anzeigen(this)">
```

Beispiel für TextRectangleObjekt:

```
<SCRIPT>
function Anzeigen(ObjektZeiger)
{
    var TextRectangleCollection = document.body.ObjektZeiger.TextRange.getClientRects();

    // .getClientRects() liefert immer Collection der textrectangle Objekte !!!

    var Rechteck = TextRectangleCollection[0];

    alert(
        "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
        + "\n"
        + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
    );
}
</SCRIPT>
<BODY>
<DIV onclick="Anzeigen(this)">
    Test
</DIV>
</BODY>
```

.leftMargin	Left Margin in Pixel des Dokumentes Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand
.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
.length	Anzahl der Zeichen im Kommentar (comment-Objekt), ab IE 6.x
.length	Anzahl der Argumente im Script-Objekt arguments als Eigenschaft eines Funktionsobjektes (siehe Script-Objekt Function und Anweisung function) Wert ist identisch mit dem Wert der Eigenschaft .length des Script-Objektes Function

Beispiel für Einlesen der Werte der Argumente aus dem Funktionsaufruf:

```
var GlobalesFeld = new Array();

function FunktionsBezeichner()
{

    // Argumenteliste der Funktion lokal zur Funktion referenzieren
    var ArgumentenListeAlsFeld = FunktionsBezeichner.arguments;

    // Anzahl der Argumente
    var ArgumenteAnzahl = ArgumentenListeAlsFeld.length;

    if ( ArgumenteAnzahl > 0 )
    {
        // Arrgumentenliste auslesen
        for (var i = 0; i < ArgumenteAnzahl; i++)
        { GlobalesFeld[i] = ArgumentenListeAlsFeld[i]; }
    }

    // hier die weiteren Anweisungen der Funktion
}
```

.length	Anzahl der Elemente in einem Feld der Objektklasse Script-Objekt Array
.length	Wert laut zeiger_auf_funktion.arguments.length siehe Script-Objekt Function und Script-Objekt arguments und Anweisung function
.length	Anzahl der Zeichen im String bzw. String-Literal Integer, ab 0 wenn 0 so Leerkette siehe Script-Objekt String

Beispiel:

```
var StringLiteral = "StringLiteral";
StringLiteral.length      liefert 13
"StringLiteral".length    liefert 13
```



.length Anzahl aller Frames bzw. IFrames im Fenster laut Collection document.frames
siehe Objekt window

.Line Anzahl der Zeilen im Textstream
sinnvoll nur verwendbar, wenn mindestens 1 newline-Zeichen in der Datei auftaucht

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.ReadAll();
alert(DateiOffen.Line);
DateiOffen.Close();
```

.link Farbe eines Links des Dokumentes
document.body.link
"#rrggbb"
vordefinierte Farbname (browserspezifisch)

.linkColor Farbe von noch nicht benutzten Links im Dokument
document.body.linkColor
"#rrggbb" Standard ist "#0000FF"
vordefinierter Farbname (browserspezifisch)

.LN2 Logarithmus zur Basis 2
siehe Script-Objekt Math

.LN10 Logarithmus zur Basis 10
siehe Script-Objekt Math

.location Zeiger auf das gleichnamige Objekt

Beispiel 1:

```
<HTML>
<HEAD>
<SCRIPT>
function LocationAendern()
{
    for(i=0;i<document.all.length;i++)
    {
        if (document.all[i].tagName=="IFRAME")
        {document.all[i].contentWindow.location = "http://www.test.com";}
    }
}
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="LocationAendern();">Location aendern</BUTTON>
<IFRAME SRC="http://www.test.de"></IFRAME>
</BODY>
</HTML>
```

Beispiel 2:

```
<HEAD>
<SCRIPT>
function Entfernen()
{
    // versuche den Text zu entfernen
    try
    {
        var KindZeigerAufTextImDiv = ID_Div.children(0);
        ID_Div.removeChild(KindZeigerAufTextImDiv);
        // Achtung: Der Text ist noch sichtbar !!!!
    }
    // oder fange das Ereignis des bereits entfernten Textes ein
    // und behandle das Ereignis
    catch(x)
    {
        alert(
            "Text wurde entfernt !\n"
            + "Das Dokument muss neu geladen werden !
        );
        document.location.reload();
    }
}
```



```

</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick=" Entfernen()">
        Klick, um diesen Text zu entfernen !
    </DIV>
</BODY>

```

Beispiel 3: Quelltext anzeigen: Öffnet lokalen Standard-Text-Editor z.B. Notepad

```
<INPUT TYPE="button" VALUE="Quelltext ansehen" onclick="window.location = 'view-source:' + window.location.href">
```

.LOG2E Logarithmus von E (eulersche Zahl) zur Basis 2
siehe Script-Objekt Math

.LOG10E Logarithmus von E (eulersche Zahl) zur Basis 10
siehe Script-Objekt Math

.logicalXDPI Standard-Anzahl der horizontalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
per screen Objekt

Beispiel 1:

```

<SCRIPT>
function ScaleFactorX()
{
    var ScaleFactor = screen.deviceXDPI / screen.logicalXDPI;
    return ScaleFactor;
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function fnScaleManually()
{
    // normale DPI-Auflösung (Dot per Inch)
    var constNorm = 96;

    // Zoomen
    if ( (screen.deviceXDPI == screen.logicalXDPI)
        && (screen.deviceXDPI > constNorm)
        )
    {
        document.body.style.zoom = constNorm / screen.logicalXDPI;
    }
}
</SCRIPT>

```

.logicalYDPI Standard-Anzahl der vertikalen Bildpunkte per Inch (DPI = Dot per Inch) auf Bildschirm
per screen Objekt

Beispiel 1:

```

<SCRIPT>
function ScaleFactorY()
{
    var ScaleFactor = screen.deviceXDPI / screen.logicalYDPI;
    return ScaleFactor;
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function fnScaleManually()
{
    // normale DPI-Auflösung (Dot per Inch)
    var constNorm = 96;

    // Zoomen
    if ( (screen.deviceYDPI == screen.logicalYDPI)
        && (screen.deviceYDPI > constNorm)
        )
    {
        document.body.style.zoom = constNorm / screen.logicalYDPI;
    }
}
</SCRIPT>

```

.longDesc Uniform Resource Identifier (URI) zur einer "long description" umwandeln

LONGTRANSITION wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software



.loop	Anzahl der Wiederholungen
	nur Objekt BG SOUND bzw. IMG mit Sound bzw. INPUT-Element mit Sound
-1	endlos
0	genau 1 mal
> 0	Anzahl der Wiederholungen
	Standard ist 1
1 entspricht	t 0

Beispiel 1:

<BGSOUND SRC="file:///c:/test/wav/test.wav">	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=0>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=1>	genau 1 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=10>	genau 10 mal
<BGSOUND SRC="file:///c:/test/wav/test.wav" LOOP=-1>	endlos

Beispiel 2:

```
<HEAD>
<SCRIPT>
    function KanalVerteilung(Wert)
    { ID_bgsound.balance = Wert;

    function GenauEinmal()
    {
        ID_bgsound.loop = 1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }

    function Endlos()
    {
        ID_bgsound.loop = -1;
        ID_bgsound.src = ID_bgsound.src; // restart
    }
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>
    <BUTTON onclick="GenauEinmal()"></BUTTON>
    <BUTTON onclick="Endlos()"></BUTTON>
    <BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung(10)"></BUTTON>
    <BUTTON onclick="KanalVerteilung (0)"></BUTTON>
    <B>Volume control:</B>&nbsp;
    <INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-10;"
    >Mute

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-7;"
    >25% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
        onpropertychange="ID_bgsound.volume=-5;"
    >50% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
    >75% Volume

    < INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
    >100% Volume
</BODY>
```

.loop	Anzahl der Scrollaktionen des marquee Objektes
0	endlos scrollen
	in Script nicht kodierbar sonst Fehlermeldung
-1	Default
	endlos scrollen
> 0	Anzahl der Scrollaktionen
	null nicht kodierbar sonst Fehlermeldung

Beispiel:

<BODY>



```

<MARQUEE BEHAVIOR="alernate" LOOP=2 onstart="alert('onstart-Ereignis erkannt')">
    Marquee Text
</MARQUEE>
</BODY>

```

.lower

Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines niedriger priorisierten Kindes
 Time-Container ist Behavior-Objektes .style.time2.excl
 Folge der in HTML kodierten Objekte t:PRIORITYCLASS entspricht der absteigenden Folge der Prioritäten der Animation.
 Jedes Kind ist ein t:PRIORITYCLASS
 darf **kein** weiteres Behavior-Objekt .style.time2.priorityClass besitzen
 wenn mehrere pausierende Kinder existieren:
 Das zuletzt pausierte Kind startet zuerst.
 Das zuerst pausierte Kind startet zuletzt.
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL ID="ID_Excl"
        BEGIN="0;indefinite;"
    >
        <t:PRIORITYCLASS>
            <SPAN ID="ID_Span1"
                CLASS="time_line_klasse"
                BEGIN="5"
                DUR="5"
            >
                Test1
            </SPAN>
        </t:PRIORITYCLASS>
        <t:PRIORITYCLASS LOWER="never">
            <SPAN ID="ID_Span2"
                CLASS="time_line_Klasse"
                BEGIN="0"
                DUR="10"
            >
                Test2
            </SPAN>
        </t:PRIORITYCLASS>
    </t:EXCL>
    <BR>
    <BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
    </BUTTON>
</BODY>
</HTML>

```

.lowsrc

Url des Images in geringerer Auflösung

.marginHeight

Abstand in Pixel vom unteren zum oberen Rand des Objektes
 Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

.marginWidth

Abstand in Pixel vom linken zum rechten Rand des Objektes
 Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

.maxLength

Maximale Anzahl der durch User eingebaren Zeichen in einem Text-Control

MAX_VALUE

maximaler numerischer endlicher Wert > 0 oder < 0, den Javascript zulässt: 1.79E+308
 Werte über MAX_VALUE sind unendlich (Number.POSITIVE_INFINITY bzw.
 Number.NEGATIVE_INFINITY)

kann als Wert zugewiesen werden
 siehe Script-Objekt Number
 nur lesen



.media Media-Typ für Ausgabe eines Objektes
 "screen" Ausgabe auf Bildschirm
 "print" Ausgabe auf Printmedium oder Druckvorschau auf Bildschirm
 "all" Default
 Ausgabe auf alle Geräte

.mediaDur Länge der Media-Datei in Sekunden (Länge bei der Wiedergabe) auf der Timeline
 Ereignis onmediacomplete muss ausgelöst also die Media-Datei komplett geladen worden sein
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:VIDEO ID="ID_Video"
SRC="test.avi"
SYNCBEHAVIOR="locked"
>
</t:VIDEO>
<SPAN ID="ID_Span"
CLASS="time_line_klasse"
DUR="0.1"
REPEATCOUNT="indefinite"
onrepeat="ID_Span.innerText= ID_Video.mediaDur;"
>
</SPAN>
</BODY>
</HTML>
```

.mediaHeight aktuelle Höhe des Media-Elementes in Pixels auf der Timeline
 Standardgemäß wird die Elementhöhe aus der Höhenangabe in der Media-Datei verwendet
 (ansonsten STYLE-Angabe zu .style.height im Element kodieren)
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:VIDEO ID="ID_Video"
SRC="test.avi"
SYNCBEHAVIOR="locked"
>
</t:VIDEO>
<SPAN ID="ID_Span"
CLASS="time_line_klasse"
DUR="0.1"
REPEATCOUNT="indefinite"
onrepeat="ID_Span.innerText= ID_Video.mediaHeight;"
>
</SPAN>
</BODY>
</HTML>
```

.mediaWidth aktuelle Breite des Media-Elementes in Pixels auf der Timeline
 Standardgemäß wird die Elementhöhe aus der Breitenangabe in der Media-Datei verwendet
 (ansonsten STYLE-Angabe zu .style.width im Element kodieren)
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
```



```

<t:VIDEO          ID="ID_Video"
                  SRC="test.avi"
                  SYNCBEHAVIOR="locked"
>
</t:VIDEO>
<SPAN ID="ID_Span"
      CLASS="time_line_klasse"
      DUR="0.1"
      REPEATCOUNT="indefinite"
      onrepeat="ID_Span.innerText= ID_Video.mediaWidth;"
>
</SPAN>
</BODY>
</HTML>

```

.menuArguments Zeiger auf dasjenige offene Fenster, indem ein Eintrag aus dem Kontextmenü ausgeführt wurde
siehe Objekt `window.external`

Beispiel:

```

<SCRIPT LANGUAGE = "JavaScript">
    var ZeigeraufWindow                                = window.external.menuArguments;

    // beispielsweise einen selektierten Text im Fenster verarbeiten
    var ZeigerAufDokuemntImWindow                      = ZeigeraufWindow.document;
    var SelektionImDokument                           = ZeigerAufDokuemntImWindow.selection;
    var TextBereichImDokument                         = SelektionImDokument.createRange();
    var SelektierterTextImTextBereich                 = TextBereichImDokument.text;

    if (SelektierterTextImTextBereich.length == 0)
    { TextBereichImDokument.text = "Einfuege Text"; }
    else
    { TextBereichImDokument.text = SelektierterTextImTextBereich.toUpperCase(); }
</SCRIPT>

```

.message Beschreibung des Run-Time-Errors
identisch mit Eigenschaft `.description`, die aber deprecated ist
siehe `error` JScript-Objekt

Beispiel:

```

function AlterErmitteln(Wert)
{
    if(Wert < 0)
    { throw new Error("Fehler: Altersangabe muss >= 0 sein"); }
}

try
{ AlterErmitteln (-5); } // aktiviert throw
catch(e)
{ alert (e.message); }

```

.method Methode des Sendens/Empfangens der Daten an/von den Server bei Formular

- "get" Daten vom Server holen und an die Url laut Eigenschaft `.action` anhängen und Url öffnen, wenn
Url ein Anker ist
- Url und angehängte Daten max. 2048 Bytes
- "post" Daten an Server senden per HTTP-Post-Transaktion
physischer Umfang der Daten ist unbegrenzt

Beispiel für Festplattenverzeichnis unter Windows auslesen:

```

<BODY>
<FORM ACTION="file:///C:/"
      METHOD="GET"
>
    <INPUT TYPE="submit" VALUE="Root von LW C anzeigen!">
</FORM>
</BODY>

```

.Methods Liste der vom Objekt unterstützten HTTP-Methoden

.mimeType MIME-Typ (Multipurpose Internet Mail Extension-Typ) des Media-Elementes auf der Timeline
siehe Objekt `currTimeState` und `Behavior` `.style.time2`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>

```



```

.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
  <t:VIDEO          ID="ID_Video"
                    SRC="test.avi"
                    SYNCBEHAVIOR="locked"
  >
  </t:VIDEO>
  <SPAN ID="ID_Span"
        CLASS="time_line_klasse"
        DUR="0.1"
        REPEATCOUNT="indefinite"
        onrepeat="ID_Span.innerText= ID_Video.mimeType;"
  >
  </SPAN>
</BODY>
</HTML>

```

.mimeType	Zeiger auf Collection mimeType siehe Objekt navigator
MIN_VALUE	minimaler numerischer endlicher Wert > 0, den Javascript zulässt: 5E-324 Werte unter MIN_VALUE sind immer 0 kann als Wert zugewiesen werden siehe Script-Objekt Number nur lesen
.mode	Ergebnis der Animation des Übergangsfilters per .style.time2.transitionFilter Behavior-Objekt, also Sichtbarkeit bzw. Unsichtbarkeit nach Vollendung des Filters siehe Objekt currTimeState und Behavior .style.time2 "in" Default Übergang hinein mit fortschreitender Animation wird Element sichtbarer, das nach der Animation voll sichtbar sein soll "out" Übergang hinaus mit fortschreitender Animation wird Element unsichtbarer, das nach der Animation nicht mehr sichtbar sein soll

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
  <t:TRANSITIONFILTER ID="ID_Transfilter1"
                      TYPE="barWipe"
                      DUR="3"
                      TARGETELEMENT="ID_Div1"
                      MODE="in"
  >
  </t:TRANSITIONFILTER>

  <t:TRANSITIONFILTER ID="ID_Transfilter2"
                      TYPE="barWipe"
                      DUR="3"
                      TARGETELEMENT="ID_Div2"
                      MODE="out"
  >
  </t:TRANSITIONFILTER>

  <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="indefinite"
        STYLE="position:relative; left:20px; width:420px; height:100px;
              background-image:url(test.gif); background-repeat: no-repeat;
              "
  >
  </DIV>

  <DIV ID="ID_Div2"

```




```

        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="indefinite"
        STYLE= "position:relative; left:20px; width:420px; height:100px;
                background-image:url(test.gif); background-repeat: no-repeat;
                "
    >
</DIV>
</BODY>
</HTML>

```

MODULATE wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software

MOTIFNAME wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software

.multiple	Multiple Auswahl bei document.select
false	Default keine multiple Auswahl
true	multiple Auswahl möglich

Beispiel:

```
<SELECT ID="ID_select" MULTIPLE>
  <OPTION>Auswahl 1</OPTION>
  <OPTION> Auswahl 2</OPTION>
  <OPTION> Auswahl 3</OPTION>
</SELECT>
<BUTTON onclick="ID_select.multiple=false">keine multiple Auswahl</BUTTON>
<BUTTON onclick="ID_select.multiple=true">multiple Auswahl</BUTTON>
```

<code>.mute</code>	Audio aktiv oder aus (stumm) auf der Timeline wenn Eigenschaft für body Objekt kodiert, so alle Audio-Elemente im Body davon betroffen beeinflusst nicht die Systemlautstärke-Regelung von Windows oder anderen Soundmixern siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code> siehe Objekt <code>bgSound</code>
<code>false</code>	Default Sound ist an
<code>true</code>	Sound ist aus

Beispiel für Abspielen einer Sounddatei per Timeline als Alternative zum Objekt bgsound:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

<t:MEDIA
ID="ID_Media"
BEGIN="indefinite;"
SRC="test.wmv"
FILL="remove"
onmediacomplete="Timer2.beginElement();"
>
</t:MEDIA>

<BR>

<BUTTON onclick="ID_Media.beginElement();">Start</BUTTON>
<BUTTON onclick="ID_Media.endElement();">Stop</BUTTON>

<BR>

<B>Volume control:</B>&nbsp;

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='true';"
>Mute

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
onpropertychange="ID_Media.mute='false'; ID_Media.volume=25;"
>25% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
onpropertychange="ID_Media.mute='false'; ID_Media.volume=50;"
```



>50% Volume

```
< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=75;"
>75% Volume
```

```
< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=100;"
>100% Volume
```

</BODY>

</HTML>

.name Name des Objektes (nicht ID !!!)
muss beim Formular für alle zu sendenden Felder kodiert sein !!
Element darf nicht per Methode .createElement() erzeugt worden sein

.name Informationen diverser Arten per meta Objekt
der Wert der Information wird in der Eigenschaft .content abgelegt

Beispiele:

Quelltextdokumentation

```
<META NAME="ProgID" CONTENT="word.document">
<META NAME="Template" CONTENT="C:\test\html.dot">
```

Anzahl der Tage nach deren Ablauf Suchmaschine erneut scannen darf

```
<META NAME="Revisit-After" CONTENT="20 days">
```

days ist festkodiert !!!

Autor

```
<META NAME="Author" CONTENT="Ich">
```

Datum des Dokumentes

```
<META NAME="Date" CONTENT="Mittwoch, 26. April 2000">
```

Dokumentbeschreibung

```
<META NAME="Description" CONTENT="freier text der von suchmaschinen indiziert wird">
```

Schlagwortliste

```
<META NAME="Keywords" LANG="xx" CONTENT="ich, du, er, sie, es, wir, ihr sie">
```

Hinweis: für xx bitte einsetzen	en	für Englisch
	de	für Deutsch
	fr	für Französisch

.name Name des MediaItem Objektes
siehe Behavior .style.mediaBar

.name Ausnahmetyp des Run-Time-Error liefern
siehe error JScript-Objekt
privater Run-Time-Error
"Error"
Standard-Run-Time-Error
"ConversionError" Konvertierungsfehler
"RangeError" Bereichfehler z.B. Feldlänge negativ
"ReferenceError" Referenz-Fehler z.B. Feld mit negativer Anzahl von Feldelementen
soll erzeugt werden
"RegExpError" Fehler bei regular Expression (RegExp-Objekt)
"SyntaxError" Typfehler z.B. bei Wertzuweisung
"TypeError" falscher Uniform Resource Indicator (URI) z.B. bei encode()
"URIError"

Beispiel:

```
function AlterErmitteln(Wert)
{
    if(Wert < 0)
    {throw new Error("Fehler: Altersangabe muss >= 0 sein");}
}

try
{AlterErmitteln (-5);} // aktiviert throw
catch(e)
{alert (e.message + " " + e.name);}
```

.name physischer Fenstertitel laut open()
entspricht Wert des Attributes TARGET eines Links
Text des Fenstertitels (Text in Titelzeile des Fensters) laut document.title
siehe Objekt window



Beispiel:

```
window.name="MyWindow";
window.open("file.htm","Frame1");
```

.Name Ordnerbezeichner in voller Länge (nicht 8.3 Bezeichner)

Beispiel:

```
var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Name);
```

.Name Dateibezeichner in voller Länge (nicht 8.3 Bezeichner)

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.Name);
```

.nameProp Dateiname-Teil einer Url ohne Path und ohne Protokoll liefern

Beispiel:

```
<SCRIPT>
function Init()
{ ID_A.innerText= ID_A.nameProp; } // test.img

window.onload=Init; // ohne () da sonst sofort ausgeführt
</SCRIPT>
<A ID="ID_A" HREF="http://www.test.de/test.img "></A>
```

NaN

nicht numerischer Wert (Not a Number)
kann als Wert zugewiesen werden
Bsp: var Wert = Number.NaN;
Hinweis: Methoden parseFloat() und parseInt() liefern NaN, wenn Parameter nicht numerisch ist.
Vergleich mit NaN ist nicht zulässig: Dafür ist die Methode .isNaN() zu verwenden.
siehe Script-Objekt Number
nur lesen

.navigator Zeiger auf das Objekt navigator
siehe Objekt window

NEGATIVE_INFINITY

entspricht Negativ-Unendlich
kann als Wert zugewiesen werden
Hinweis für numerische Operationen

Multiplikation	Wert mal unendlich ergibt unendlich unendlich mal unendlich ergibt unendlich 0 * unendlich ergibt NaN NaN * unendlich ergibt NaN
Division	Wert durch unendlich ergibt 0 unendlich durch Wert ergibt unendlich unendlich durch unendlich ergibt NaN Division durch 0 nicht erlaubt
Vorzeichen	wird wie üblich ermittelt

siehe Script-Objekt Number
nur lesen

.nextItem Zeiger auf nächsten Eintrag in der Playliste (Objekt PlaylistInfo)
siehe Behavior .style.mediaBar

Beispiel:

```
<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick= "ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
ID_Div.enabled = false;
"
>
```

.nextSibling Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes

Beispiel:

```
<SCRIPT>
```



```

var ErstesKind_Index = 0;      // Index ab 0
var ZeigerAufNachfolgerKind = Liste.childNodes(ErstesKind_Index).nextSibling;
                                // liefert Referenz auf Listenelement 2

```

```

</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1
    <LI> Listenelement 2
    <LI> Listenelement 3
  </UL>
</BODY>

```

.nodeName String als Name des Kindes (Knoten, Node, Element)
 also TAG-Bezeichner, Attribut-Name; #text für Anker

Beispiel:

```

<SCRIPT>
var ErstesKind_Index = 0;      // Index ab 0
var ErstesKind_Name = Liste.childNodes(ErstesKind_Index).nodeName;
alert(ErstesKind_Name);       // liefert den Tagnamen 'LI' von Listenelement 1
                                // Hinweis: Listenelement 1 ist der Wert des Kindes
</SCRIPT>
<BODY>
  <UL ID = "Liste">
    <LI> Listenelement 1
    <LI> Listenelement 2
    <LI> Listenelement 3
  </UL>
</BODY>

```

.nodeType Knotentyp laut attributes Collection

1	für Element-Knoten
2	für Attribut-Knoten
3	für Textknoten
4	für CDATA-Abschnitt
5	für Entity-Referenz
6	für Entity-Knoten
7	für Processing Instruction
8	für Kommentar-Knoten
9	für das Dokument
10	für den Dokumenttyp
11	für ein Dokument-Fragment
12	für Notation

Beispiel 1:

```
var KnotenTypVonBODY = document.body.nodeType;
```

Beispiel 2:

```

var ZeigerAuf_B_Tag = document.createElement("B");      // B-Tag erzeugen
document.body.insertBefore(ZeigerAuf_B_Tag);
                                // B-Tag in den BODY-Teil des Dokument einfügen,
                                // also in die Baumstruktur
var KnotenTypDes_B_Tag = ZeigerAuf_B_Tag.nodeType;

```

.nodeValue Knotenwert (Wert des Kindes, Node, Elementes)
 nur für Text- und Attribut-Elemente
 nicht für Element-Knoten (Knotentyp 1)

Beispiel:

```

<SCRIPT>
function KnotenWertAendern( ZeigerAufListe,
                             IndexVonListenElement,      // immer ab 0
                             Zeichenkette                 // Listenelement muss Text sein
                           )
{
    var ReturnWert=false;      // Annahme: Änderung schlägt fehl

    // prüfen auf UL-Tag
    if (ZeigerAufListe.nodeName == "UL")
    {
        // Anzahl der Listenelemente holen
        var AnzahlListenelemente= ZeigerAufListe.childNodes.length;
                                                // immer ab 1

        // und Anzahl und Index prüfen
        if ( (AnzahlListenelemente > 0)      // immer ab 1
            && (IndexVonListenElement >= 0)   // immer ab 0
            && (IndexVonListenElement < AnzahlListenelemente)
        )
        {
            // ...
        }
    }
}

```



```

    )
    {
        // Zeiger auf das Listenelement laut Index holen
        var ZeigerAufListenElement =
            ZeigerAufListe.childNodes(IndexVonListenElement);

        // existiert das Listenelement ?
        if (ZeigerAufListenElement)
        {
            // ZeigerAufListenElement ist nicht null

            // Listenelement ist Textelement ?
            if (ZeigerAufListenElement.nodeType == 3)
            {
                ZeigerAufListenElement.nodeValue =
                    Zeichenkette;
                ReturnWert = true;
            }
        }
    }

    return ReturnWert;
}
</SCRIPT>
<UL ID="Liste" onclick=" KnotenWertAendern(this, 0, 'Listenelement Neu')">
    <LI>Listenelement alt
</UL>

```

.noHref HREF-Wirkung (Eigenschaft .href) ein/ aus bei Click auf den Link
 false Default
 Link ist wirksam bei click
 true Link ist unwirksam bei click

.noResize Frame-Größenänderung ein/aus
 Größenänderung z.B. durch Maus etc.
 per frame Objekt
 false Default Größe änderbar
 true Größe nicht änderbar

.noWrap Wortumbruch einstellen
 false Default.
 Browser bricht den Text automatisch um
 true Browser bricht den Text nicht um

.number Nummer des Run-Time-Error
 siehe error JScript-Objekt

Beispiel:

```

function AlterErmitteln(Wert)
{
    if(Wert < 0)
    {throw new Error(8888,"Fehler: Altersangabe muss >= 0 sein");}
}

try
{AlterErmitteln (-5);} // aktiviert throw
catch(e)
{alert (e.message + " " + e.number);}

```

.object Zeiger auf das Objekt laut Applet bzw. laut Objekt object

.offscreenBuffering alle Objekt in aktueller Seite im Offscreen zeichnen bevor sie sichtbar werden
 siehe Objekt window

.offsetHeight Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
 des Elternobjektes (.offsetParent)

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    var Faktor = 4;

    function startClock()

```



```

    {
        window.setInterval("Clock_Tick()", 1000);
        runClock();
    }

function runClock()
{
    var DatumHeute = Date();
    var ZeitHeute = DatumHeute.substring(11,19);
    var ElternHoehe = document.body.offsetHeight;
    var ElternBreite = document.body.offsetWidth;

    if ( (ElternHoehe * Faktor) > ElternBreite )
    { ElternHoehe = ElternBreite / Faktor; }

    document.all. ID_P.innerText = ZeitHeute;
    document.all. ID_P.style.fontSize = ElternHoehe;
}
</SCRIPT>
</HEAD>
<BODY onload="startClock()">
    <P ID="ID_P">&nbsp;</P>
</BODY>
</HTML>

```

Beispiel 2:

```

<DIV ID="ID_Div" STYLE="overflow:scroll; width:200; height:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientHeight)">client height</BUTTON>
<BUTTON onclick="alert(ID_Div.offsetHeight)">offset height</BUTTON>

```

.offsetLeft X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (**.offsetParent**)

Beispiel:

```

<SCRIPT>
function Anzeige(ObjektZeiger oObject)
{
    var ElternZeiger = ObjektZeiger.offsetParent;
    var ElternBreite = ElternZeiger.clientWidth;

    var KindOffsetLeft = ID_Div.offsetLeft;

    if ( KindOffsetLeft > ElternBreite )
    { alert("nach rechts scrollen"); }
}
</SCRIPT>
<BUTTON onclick="Anzeige(this)">Klick mich</BUTTON>
<DIV ID="ID_Div" STYLE="position:absolute; top:200; left:1200;"> </DIV>

```

.offsetParent Referenz der Eltern für Nutzung von **.offsetHeight**, **.offsetLeft**, **.offsetTop** und **.offsetWidth**

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    function Anzeige()
    {
        var ZeigerAufKind = document.all.ID_TD;

        alert( "TD liegt an Position (Left, Top) ("
            + ZeigerAufKind.offsetLeft
            + ", "
            + ZeigerAufKind.offsetTop
            + ")\n"
            + "Eltern als Container von TD ist "
            + ZeigerAufKind.offsetParent.tagName
        );
    }
</SCRIPT>
</HEAD>
<BODY onload="Anzeige()">
    <TABLE BORDER=1 ALIGN=right>
        <TR>
            <TD ID="ID_TD" >TD als Kind</TD>

```



```

        </TR>
    </TABLE>
</BODY>
</HTML>

```

.offsetTop Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeige(ObjektZeiger)
    {
        var ElternZeiger = ObjektZeiger.offsetParent;
        var ElternHoehe = ElternZeiger.clientHeight;

        var KindOffsetTop = ObjektZeiger.offsetTop;

        if (KindOffsetTop > ElternHoehe)
        {alert("Testtext nicht sichtbar. Fenster erweitern für Sichtbarkeit");}
        else
        {alert("Testtext ist sichtbar");}
    }
</SCRIPT>
</HEAD>
<BODY onload="window.resizeTo(430,250)" onclick="Anzeige(ID_Div)" SCROLL=NO>
<DIV STYLE="position:absolute;left:20px">
    Irgendwo in das Fenster klicken
</DIV>
<DIV ID="ID_Div" STYLE=
    "position:absolute;"
    + "left:50px;top:300px;width:280px;color:lightGreen;"
    + "font-size:large;font-weight:bold;"
    + "background-color:hotPink;font-family:Arial"
>
    Testtext
</DIV>
</BODY>
</HTML>

```

.offsetWidth X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    var Faktor = 4;

    function startClock()
    {
        window.setInterval("Clock_Tick()", 1000);
        runClock();
    }

    function runClock()
    {
        var DatumHeute = Date();
        var ZeitHeute = DatumHeute.substring(11,19);
        var ElternHoehe = document.body.offsetHeight;
        var ElternBreite = document.body.offsetWidth;

        if ( (ElternHoehe * Faktor) > ElternBreite )
        { ElternHoehe = ElternBreite / Faktor;}

        document.all. ID_P.innerText = ZeitHeute;
        document.all. ID_P.style.fontSize = ElternHoehe;
    }
</SCRIPT>
</HEAD>
<BODY onload="startClock()">
    <P ID="ID_P">&nbsp;&nbsp;&nbsp;</P>
</BODY>
</HTML>

```



Beispiel 2:

```
<DIV ID="ID_Div" STYLE="overflow:scroll; width:200; height:100">Test</DIV>
<BUTTON onclick="alert(ID_Div.clientWidth)">client width</BUTTON>
<BUTTON onclick="alert(ID_Div.offsetWidth)">offset width</BUTTON>
```

.offsetX

X-Koordinate Maus relativ zum Objekt, das das Event auslöst
Objekt event

Beispiel:

```
<HEAD>
<SCRIPT>
function AlertAnzeige()
{
    var Kette = "X= " + window.event.offsetX + "\r";
    Kette += "Y= " + window.event.offsetY + "\r";
    alert(Kette);
}

function StatusZeileAnzeigen()
{
    var Kette = "X= " + window.event.offsetX + " ";
    Kette += "Y= " + window.event.offsetY;
    window.status = Kette;
}
</SCRIPT>
</HEAD>
<BODY>
<DIV STYLE="position:absolute;top:200px;left:300px;"
onmousemove="StatusZeileAnzeigen()" ondblclick="AlertAnzeige()">
</DIV>
</BODY>
```

.offsetY

Y-Koordinate Maus relativ zum Objekt, das das Event auslöst
Objekt event

Beispiel:

```
<HEAD>
<SCRIPT>
function AlertAnzeige()
{
    var Kette = "X= " + window.event.offsetX + "\r";
    Kette += "Y= " + window.event.offsetY + "\r";
    alert(Kette);
}

function StatusZeileAnzeigen()
{
    var Kette = "X= " + window.event.offsetX + " ";
    Kette += "Y= " + window.event.offsetY;
    window.status = Kette;
}
</SCRIPT>
</HEAD>
<BODY>
<DIV STYLE="position:absolute;top:200px;left:300px;"
onmousemove="StatusZeileAnzeigen()" ondblclick="AlertAnzeige()">
</DIV>
</BODY>
```

.onLine

Onlinestatus des Browsers ermitteln per navigator Objekt
keine Überprüfung auf Netzwerkverbindung
navigator.onLine
true ist offline
false ist online

.onLine

System-Online-Status (nicht Netzwerk-Online-Status !!!)
siehe Objekt window.clientInformation
true ist offline
false ist online
Hinweis: Eigenschaft .connectionType des Behavior clientCaps zeigt den Status der
genutzten Verbindung an

.onOffBehavior

deprecated ab IE 5.x
Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound



.opener Zeiger auf Elternfenster, das **open()** enthält und das Kind-Fenster öffnet, welches in dieser **.opener**-Eigenschaft den Zeiger auf das Elternfenster enthält
 Bezug im geöffneten Fenster auf Instanzen des Aufrufers ist möglich
 Bezug des Aufrufers auf geöffnetes Fenster ist möglich
 Hinweis: Bei FRAME und IFRAME anstelle opener den Zeiger parent verwenden
 Öffnet ein Frame / IFrame ein Fenster, das damit nicht im Frameset läuft, so ist der Bezug vom Fenster aus auf das Frameset oder auf einen Frame im Frameset per **window.opener.parent** möglich, solange opener existiert.
 siehe Objekt window

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
        + '<HEAD></HEAD>'
        + '<BODY>'
        + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
        + '<BR>'
        + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
        + ' onclick="opener.OnClickHandler();"'
        + '>'
        + '</BODY>'
        + '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.openState Status des Media Bar Player bezüglich Playlist, Codec, Lizenz und Individualisierung
 Media-Datei kann haben
 Codec (Ländernummer)
 Lizenz bei käuflichem Erwerb durch den User
 Individualisierung auf den User
 Media Bar Player ist der Windows Media Playersiehe Behavior .style.mediaBar

Beispiel:

```
<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
onopenstatechange="alert(ID_Div.openState);"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
"
>
```

.options Zeiger auf die Collection document.select.options des document.select Objektes
 Feld der Zeiger aller document.select.option Objekte

.origin Bezugspunkt der Animation eines Elementes auf der Timeline
 nur für Objekt animatemotion (t:ANIMATEMOTION)
und .additive muss "replace" sein
 siehe Objekt currTimeState und Behavior .style.time2

.outerHTML Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag
 wirksam mit parsen des Ende-Tag
 nur nach kompletten Einlesen des Dokumentes nutzbar
 schreiben: wenn Bereich gefüllt so komplett überschreiben
 Plain-Text
 HTML-Elemente möglich
 nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.

Beispiel:

```
<SCRIPT>
```



```

function ListeAendern()
{
    var ListenElement = event.srcElement;

    if ( (ListenElement.tagName != "UL")
        && (ListenElement.tagName != "LI")
        )
    {
        alert("ListenElement.outerHTML + " zur Liste hinzufügen");
        ID_Div.innerHTML += ListenElement.outerHTML + "<BR>";
    }
}
</SCRIPT>
<UL onclick = " ListeAendern()">
    <LI><B>Listenelement 1</B>
    <LI><I> Listenelement 2</I>
    <LI><U> Listenelement 3</U>
    <LI><STRIKE> Listenelement 4</STRIKE>
</UL>
<DIV ID="ID_Div"></DIV>

```

.outerText Referenz auf den gesamten Plain-Text im Objekt
 nur nach kompletten einlesen des Dokumentes nutzbar
 nur Plain-Text
 schreiben: immer komplett überschreiben
 nur lesen bei HTML, THEAD, TBODY, TFOOT, TD, TH, TR

Beispiel:

```

<DIV ID="ID_Div">
    <P ID="ID_P">Dieser Text wird veraendert</P>
</DIV>
<BUTTON onclick="ID_P.outerText='ändern'">
    Aendern
</BUTTON>
<BUTTON onclick="ID_Div.innerHTML='<P ID=ID_P>Dieser Text wird veraendert</P>'">
    Reset
</BUTTON>

```

.ownerDocument Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde

Beispiel:

```

<SCRIPT>
    var ElternDokument = SpanTag.ownerDocument;
</SCRIPT>
<BODY>
    <SPAN ID="SpanTag">Hallo !</SPAN>
</BODY>

```

.owningElement Referenz auf den im StyleSheet implementierten Style als Kindobjekt
 Hinweis: Es sind die Eigenschaften und Methoden des style Objektes referenzierbar
 siehe Objekt styleSheet

Beispiel:

```

for ( var i = 0; i < document.styleSheets.length; i++ )
{
    if ( document.styleSheets(i).owningElement.tagName == "STYLE" )
    {
        for ( var j = 0; j < document.styleSheets(i).imports.length; j++ )
        {
            alert( "Importierter StyleSheet (Style-Regel) Nr " + j
                + " hat HREF = " + document.styleSheets(i).imports(j).href
            );
        }
    }
}

```

.parent Zeiger auf das **in der Fensterhierarchie übergeordnete** Fenster, das aber nicht das .open() zum
 Kindfenster enthalten muss
 z.B. Zeiger auf das Fenster, das die FRAMESET-Deklaration enthält,
 oder das diesem Fenster übergeordnet ist
 Test auf Existenz von parent per if (parent != self)
 siehe Objekt window

.parentElement Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
 var Zeiger = document.body.parentElement;



.ParentFolder Bezeichner des Eltern-Ordners liefern, in dem der Ordner liegt

Beispiel 1:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ParentFolder);
```

Beispiel 2:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
var Zahler          = 0;
if (!Ordner.IsRootFolder)
{
    do
    {
        Ordner = Ordner.ParentFolder;
        Zahler++;
    }
    while (!Ordner.IsRootFolder);
}

alert("Ordner liegt " + Zahler + " Ebenen unter der Root");
```

.ParentFolder Bezeichner des Ordners liefern, in dem die Datei liegt

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.ParentFolder);
```

.parentNode Referenz auf Elternknoten innerhalb der DOM-Hierarchie

Beispiel 1:

```
<SCRIPT>
    var ElternZeiger = SpanTag.parentNode;
</SCRIPT>
<BODY>
    <SPAN ID="SpanTag">Hallo !</SPAN>
</BODY>
```

Beispiel 2:

document.body.parentNode wird null liefern, da BODY das oberste Objekt

Beispiel 3:

```
var ZeigerAuf_B_Tag = document.createElement("B");    // erzeugen
document.body.insertBefore(ZeigerAuf_B_Tag);         // einfügen
var ElternZeiger = ZeigerAuf_B_Tag.parentNode;       // Zeiger auf BODY
```

.parentStyleSheet Referenz auf das den StyleSheet implementierende Objekt (Elternobjekt)
siehe Objekt styleSheet

.parentTextEdit Textbereich des Elternobjektes referenzieren

Beispiel:

```
<SCRIPT>
    function Selektion()
    {
        var ZeigerAufSelektionsQuelle = window.event.srcElement ;

        if (!ZeigerAufSelektionsQuelle.isTextEdit)
        { ZeigerAufSelektionsQuelle = ZeigerAufSelektionsQuelle.parentTextEdit; }

        if (ZeigerAufSelektionsQuelle != null)
        {
            var ZeigerAufTextBereich =
                ZeigerAufSelektionsQuelle.createTextRange();

            ZeigerAufTextBereich.moveToElementText(window.event.srcElement);
            ZeigerAufTextBereich.collapse();
            ZeigerAufTextBereich.expand("SelektionsText");
            ZeigerAufTextBereich.select();
        }
    }
</SCRIPT>
```



.parentTimeBegin Startzeit als Offset von der Startzeit der Eltern-Timeline
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function ZeitHolen()
{
    ID_Span.innerText +=
        ID_Div.currTimeState.parentTimeBegin + ' Sekunden';
}
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Span1"
CLASS="time_line_klasse"
DUR=".01"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
>0
</SPAN>
<BR>
<SPAN ID="ID_Span2"
CLASS="time_line_klasse"
DUR=".01"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=parseInt(ID_excl.currTimeState.activeTime);"
>0
</SPAN>
<BR>
<t:EXCL ID="ID_excl "
REPEATDUR="indefinite"
onbegin=" ZeitHolen();"
>
    <DIV ID="ID_Div"
CLASS="time_line_klasse"
BEGIN="1"
DUR="5"
>
    </DIV>
</t:EXCL>
<BR>
<SPAN ID="ID_Span">
0 Sekunden
</SPAN>
</BODY>
</HTML>
```

.parentTimeEnd Endezeit als Offset von der Startzeit der Eltern-Timeline
 ist zugleich Summe der Werte laut Eigenschaften
 parentTimeBegin und activeDur
 Hinweis zu activeDur: inklusive aller Wiederholungen und Autoreverse
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function ZeitHolen()
{
    ID_Span.innerText +=
        ID_Div.currTimeState.parentTimeEnd + ' Sekunden';
}
</SCRIPT>
```



```

</HEAD>
<BODY>
  <SPAN ID="ID_Span1"
    CLASS="time_line_klasse"
    DUR=".01"
    REPEATCOUNT="indefinite"
    FILL="hold"
    onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
  >
    0
  </SPAN>
  <BR>
  <SPAN ID="ID_Span2"
    CLASS="time_line_klasse"
    DUR=".01"
    REPEATCOUNT="indefinite"
    FILL="hold"
    onrepeat="innerText=parseInt(ID_excl.currTimeState.activeTime);"
  >
    0
  </SPAN>
  <BR>
  <t:EXCL ID="ID_excl"
    REPEATDUR="indefinite"
    onbegin="ZeitHolen();"
  >
    <DIV ID="ID_Div"
      CLASS="time_line_klasse"
      BEGIN="4"
      DUR="10"
      AUTOREVERSE="true"
    >
      </DIV>
  </t:EXCL>
  <BR>
  <SPAN ID="ID_Span">
    0 Sekunden
  </SPAN>
</BODY>
</HTML>

```

.parentWindow Referenz auf Elternfenster des Dokumentes
document.parentWindow

.path Liste von Werten für 2D-Vektorgraphik-Animation eines Elementes auf der Timeline
nur für Objekt `animatemotion` (t:ANIMATEMOTION)
siehe `.values` für 2D-Animation anhand absoluter Koordinaten im Grafiksystem
wenn `.calcMode` auf "paced" so werden Move To-Kommandos ignoriert
für die Objekte `animate`, `animateMotion` und `animateColor` gilt:
 `.path` wird von keiner Eigenschaft überschrieben
 `.path` überschreibt `.by` `.from` `.to` `.values`
Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel 1:

path="M 0 0 L 100 0 v 100 h -100 V 0"

erreichte Position absolut zum Ursprung 0,0			
Operation	X-Ordinate	Y-Ordinate	
Pfadanfang	0	0	
Line ziehen	100	0	
vertikale Linie	100	100	um 100 Pixel nach oben +100
horizontale Linie	0	100	um 100 Pixel nach links - 100
vertikale Linie	0	0	

Hinweis: Anstelle von "V 0" hätte auch Z oder z kodiert werden können.

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

```



```

<DIV ID="ID_Div1"
      CLASS="time_line_klasse"
      STYLE="position:absolute;top:50px;left:50px;height:100px;width:150px;
            border:solid black 1px;
            "
>
    animierter Div mit Vektorgraphik
</DIV>
<t:ANIMATEMOTION TARGETELEMENT="ID_Div1"
                  BEGIN="ID_Button.click"
                  PATH="M 0 0 L 100 0 v 100 h -100 V 0"
                  DUR="3"
>
</t:ANIMATEMOTION>
<DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      STYLE="position:absolute;top:50px;left:250px;height:100px;width:150px;
            border:solid black 1px;
            "
>
    animierter Div mit Vektorgraphik
</DIV>
<t:ANIMATEMOTION TARGETELEMENT="ID_Div2"
                  BEGIN="ID_Button.click"
                  PATH="m 0 0 L 100 0 100 100 0 100 z"
                  DUR="3"
>
</t:ANIMATEMOTION>
<BUTTON ID="ID_Button">Starten/Restarten</BUTTON>
</BODY>
</HTML>

```

.Path Pfad des Ordners liefern in voller Länge (nicht 8.3 Pfad)

Beispiel:

```

var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Path);

```

.Path Pfad der Datei liefern in voller Länge (nicht 8.3 Pfad)

Beispiel:

```

var DateinameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.Path);

```

.Path Pfad eines Laufwerkes ermitteln in voller Länge (nicht 8.3 Pfad)

Beispiel:

```

var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk         = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Pfad    = Laufwerk.Path;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Pfad);

```

.pathname Datei und Pfad eines Objektes

.peers

Abbruch bzw. Pausierung der aktiven Animation durch Start der Animation eines Kindes
Time-Container ist Behavior-Objektes .style.time2.excl

Es gibt **genau** ein t:PRIORITYCLASS, in dem **kein** weiteres Behavior-Objekt
.style.time2.priorityClass kodiert sein darf.

wenn mehrere pausierende Kinder existieren:

Das zuletzt pausierte Kind startet zuerst.

Das zuerst pausierte Kind startet zuletzt.

siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

```




```

<t:EXCL ID="ID_Excl"
  BEGIN="0;indefinite;"
>
  <t:PRIORITYCLASS PEERS="pause">
    <SPAN ID="ID_Span1"
      CLASS="time_line_Klasse"
      BEGIN="0"
      DUR="10"
    >
      Test1
    </SPAN>
    <SPAN ID="ID_Span2"
      CLASS="time_line_Klasse"
      BEGIN="5"
      DUR="3"
    >
      Test2
    </SPAN>
  </t:PRIORITYCLASS>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.beginElement();">
</BUTTON>
</BODY>
</HTML>

```

.PI Zahl Pi
 siehe Script-Objekt Math

.platform Name des Betriebssystems per navigator Objekt
navigator.platform
"HP-UX" HP Unix
"MacPPC" Macintosh PowerPC
"Mac68K" Macintosh 68K
"SunOS" Solaris
"Win32" Microsoft Windows 32-Bit
"Win16" Microsoft Windows 16-Bit
"WinCE" Microsoft Windows CE

Beispiel für prüfen auf ActiveX beim IE:

```

var ActiveXAktiv = false;

var NavigatorObjekt = window.navigator;                      // Zeiger

var BrowserArt      = NavigatorObjekt.userAgent;            // String
var IEerkannt      = (BrowserArt.indexOf('IE') > -1);        // true, so IE erkannt

var Plattform       = NavigatorObjekt.platform;              // String
var Win32Erkannt   = (Plattform == "Win32");                // true, so Win32-Bit erkannt

ActiveXAktiv       = (IEerkannt && Win32Erkannt);           // true, so IE und Win32-Bit erkannt,
                                                                 // also ActiveX möglich

```

.platform Windows-Betriebssystem
Behavior .style.clientCaps
"Win32" Windows 32-Bit
"Win16" Windows 16-Bit
"WinCE" Windows CE

.platform Betriebssystem
siehe Objekt window.clientInformation
"HP-UX" HP Unix
"MacPPC" Macintosh PowerPC
"Mac68K" Macintosh 68K
"SunOS" Solaris
"Win32" Microsoft Windows 32-Bit
"Win16" Microsoft Windows 16-Bit
"WinCE" Microsoft Windows CE

.player Wiedergabeplayer einem Media-Element zuordnen für Wiedergabe auf der Timeline
Player ist eine Softwarekomponente, die im System und im Browser installiert sein muss
ab IE 6.x werden keine Plugins mehr unterstützt sondern nur noch ActiveX-Controls
z.B. ist {6BF52A52-394A-11d3-B153-00C04F79FAA6} das ActiveX-Control für den
Windows Media-Player 7.1



siehe Objekt currTimeState und Behavior .style.time2

Beispiel für Wiedergabe per Media Bar-Player:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
  <t:PAR SYNCBEHAVIOR="locked">
    <t:MEDIA ID="ID_Media"
      SRC="test.mid"
      PLAYER="{52ca3bcf-3b9b-419e-a3d6-5d28c0b0b50c}"
      SYNCMASTER="true"
      SYNCBEHAVIOR="locked"
      BEGIN="1"
    >
    </t:MEDIA>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
      BEGIN="ID_Media.begin"
      DUR="9"
      TARGETELEMENT="ID_Div1"
      PATH="M 0 0 L 100 300"
      FILL="freeze"
    >
    </t:ANIMATEMOTION>
    <t:SEQ ID="ID_Seq"
      STYLE="font-size:18pt;color:#ff0000"
      SYNCBEHAVIOR="locked"
      BEGIN="ID_Media.begin"
      FILL="freeze"
    >
      <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        Diese MIDI-File
      </DIV>
      <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        wird mit dem
      </DIV>
      <DIV ID="ID_Div3"
        CLASS="time_line_klasse"
        DUR="2"
      >
        Media Bar-Player
      </DIV>
      <DIV ID="ID_Div4"
        CLASS="time_line_klasse"
        DUR="2"
      >
        abgespielt.
      </DIV>
    </t:SEQ>
  </t:PAR>
  <DIV ID="ID_Div5"
    CLASS="time_line_klasse"
    STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
      top:70;left:10;font-size:18;border-style:solid"
  >
    animierter DIV waehrend der Wiedergabe der MIDI-File
  </DIV>
</BODY>
</HTML>
```

Beispiel für Wiedergabe per Windows Media Player:



```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
  <t:PAR SYNCBEHAVIOR="locked">
    <t:MEDIA ID="ID_Media"
      SRC="test.mid"
      PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}"
      SYNCMASTER="true"
      SYNCBEHAVIOR="locked"
      BEGIN="1"
    >
  </t:MEDIA>
  <t:ANIMATEMOTION ID="ID_Animatemotion"
    BEGIN="ID_Media.begin"
    DUR="9"
    TARGETELEMENT="ID_Div1"
    PATH="M 0 0 L 100 300"
    FILL="freeze"
  >
</t:ANIMATEMOTION>
  <t:SEQ ID="ID_Seq"
    STYLE="font-size:18pt;color:#ff0000"
    SYNCBEHAVIOR="locked"
    BEGIN="ID_Media.begin"
    FILL="freeze"
  >
    <DIV ID="ID_Div1"
      CLASS="time_line_klasse"
      DUR="1.5"
    >
      Diese MIDI-File
    </DIV>
    <DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      DUR="1.5"
    >
      wird mit dem
    </DIV>
    <DIV ID="ID_Div3"
      CLASS="time_line_klasse"
      DUR="2"
    >
      Windows Media-Player
    </DIV>
    <DIV ID="ID_Div4"
      CLASS="time_line_klasse"
      DUR="2"
    >
      abgespielt.
    </DIV>
  </t:SEQ>
</t:PAR>
  <DIV ID="ID_Div5"
    CLASS="time_line_klasse"
    STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
      top:70;left:10;font-size:18;border-style:solid
      "
  >
    animierter DIV waehrend der Wiedergabe der MIDI-File
  </DIV>
</BODY>
</HTML>

```

.playerObject

Zeiger auf den Wiedergabeplayer, der das Media-Element auf der Timeline wiedergibt
 besonders zu verwenden, wenn Player über ActiveX-Control im Browser implementiert ist
 Zeiger ermöglicht, die Methoden und Eigenschaften des Players anzusprechen
 siehe Playerbeschreibung des Herstellers vom Player
 (im Falle von Microsoft sind die Methoden und Eigenschaften
 im jeweiligen SDK der Player-Version zu finden)



	<p>Playerart laut .player siehe Objekt currTimeState und Behavior .style.time2</p>						
.playlistInfo	<p>Zeiger auf die Playliste (Objekt PlaylistInfo) siehe Behavior .style.mediaBar</p>						
.playState	<p>Wiedergabe-Status des Media Bar Player (Wiedergabe der Media-Datei) Status wird verändert durch Aktionen des Users mit dem Player Media Bar Player ist der Windows Media Player Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen. Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die</p>						
aktuelle	<p>Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei. siehe Methoden .playNext() .playURL() .stop() siehe Behavior .style.mediaBar 0 undefiniert 1 Wiedergabe ist gestoppt 2 Wiedergabe pausiert 3 Player gibt gerade einen Daten-Stream wider 4 Player springt geraden den Nachfolger-Daten-Stream an 5 Player springt geraden den Vorgänger-Daten-Stream an 6 Player puffert gerade Media-Daten 7 Player wartet gerade auf Daten-Stream 8 Player hat das Ende der Media-Datei erkannt, also das Ende des letzten Daten-Stream in der Media-Datei 9 Player bereitet gerade eine neue Media-Wiedergabe vor 10 Player ist bereit für Wiedergabe</p>						
.plugins	<p>Zeiger auf Collection plugin siehe Objekt navigator</p>						
.port	<p>Port einer Location oder Url in der Form "hostname:port" basierend auf dem aktuellen Protokoll laut Eigenschaft .protocol z.B.</p> <table> <tr> <th>Standard-Ports</th><th>Protokoll</th></tr> <tr> <td>21</td><td>FTP</td></tr> <tr> <td>80</td><td>HTTP</td></tr> </table>	Standard-Ports	Protokoll	21	FTP	80	HTTP
Standard-Ports	Protokoll						
21	FTP						
80	HTTP						
Beispiel:	<pre> <SCRIPT> function getPort() { alert ("FTP: " + ID_A1.port + "\n" + "HTTP: " + ID_A2.port); } </SCRIPT> Link1 Link2 </pre>						
POSITIVE_INFINITY	<p>entspricht Positiv-Unendlich kann als Wert zugewiesen werden Hinweis für numerische Operationen</p> <table> <tr> <td>Multiplikation</td><td>Wert mal unendlich ergibt unendlich unendlich mal unendlich ergibt unendlich 0 * unendlich ergibt NaN NaN * unendlich ergibt NaN</td></tr> <tr> <td>Division</td><td>Wert durch unendlich ergibt 0 unendlich durch Wert ergibt unendlich unendlich durch unendlich ergibt NaN Division durch 0 nicht erlaubt</td></tr> <tr> <td>Vorzeichen</td><td>wird wie üblich ermittelt</td></tr> </table> <p>siehe Script-Objekt Number nur lesen</p>	Multiplikation	Wert mal unendlich ergibt unendlich unendlich mal unendlich ergibt unendlich 0 * unendlich ergibt NaN NaN * unendlich ergibt NaN	Division	Wert durch unendlich ergibt 0 unendlich durch Wert ergibt unendlich unendlich durch unendlich ergibt NaN Division durch 0 nicht erlaubt	Vorzeichen	wird wie üblich ermittelt
Multiplikation	Wert mal unendlich ergibt unendlich unendlich mal unendlich ergibt unendlich 0 * unendlich ergibt NaN NaN * unendlich ergibt NaN						
Division	Wert durch unendlich ergibt 0 unendlich durch Wert ergibt unendlich unendlich durch unendlich ergibt NaN Division durch 0 nicht erlaubt						
Vorzeichen	wird wie üblich ermittelt						
.previousSibling	<p>Referenz auf das Vorgängerkind</p>						
Beispiel:	<pre> <SCRIPT> var AktuellesKind_Index = 1; // Index ab 0 var VorgaengerKind_Zeiger = Liste.childNodes(AktuellesKind_Index).previousSibling; // Listenelement 1 wird referenziert </SCRIPT> <BODY> <UL ID="Liste"> Listenelement 1 Listenelement 2 Listenelement 3 </pre>						



<BODY>

.progress Fortschrittsangabe entlang der Timeline
 inklusive Repeats etc.
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
function update()
{
    var Fortschritt = movie.currTimeState.progress;
    var Kette = Fortschritt.toString();

    if (movie.currTimeState.stateString != "holding")
    {
        // Element wird nicht gehalten
        if (Fortschritt != 0)
        {
            if (Kette.substr(2,1) == "0")
            {
                Kette = Kette.substr(3,1); // 1 bis 9
            }
            else
            {
                Kette = Kette.substr(2,2); // 10 bis 99
            }
        }
        else
        { Kette = "0"; }
    }
    else
    {
        // Element wird gehalten
        Kette = "100";
    }

    alert(Kette + "%");
}
</SCRIPT>
```

.propertyIsEnumerable prüfen ob Stringwert in einem Objekt als Menge von String-Elementen enthalten ist
 und ob das Objekt mit der Anweisung for in verarbeitet werden kann

Beispiel:

```
var Feld = new Array("Apfel", "Banane", "Zitrone");
alert( (Feld.propertyIsEnumerable("Apfel"));                      // true
```

.propertyName Name der Eigenschaft, die wertmässig verändert wurde durch onpropertychange Event
 auch Style-Eigenschaft etc.
 Objekt event

Beispiel:

```
<HEAD>
<SCRIPT>
function ValueAendern()
{ ID_Input1.value = "Neuer Wert von VALUE";}

function FarbeAendern()
{ ID_Input2.style.backgroundColor = "aqua";}

function Anzeige()
{alert(event.propertyName + " wurde verändert");}

</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE=button ID="ID_Input1"
VALUE="Click um VALUE zu ändern"
onclick="ValueAendern()"
onpropertychange="Anzeige()"
>
<INPUT TYPE=button ID="ID_Input2"
VALUE="Click um Farbe zu ändern"
onclick="FarbeAendern()"
onpropertychange="Anzeige()"
>
</BODY>
```



.protocol Protokoll-Teil einer Url inklusive http und ftp liefern
 nur lesen bei Objekten document
 img
 location

Beispiel:

location.protocol liefert z.B. http: und immer inklusive dem Doppelpunkt

.prototype Zeiger auf den Prototyp-Bereich im Objekt, der per Prototyping erweitert wird
 Objekt ist entweder Script-Objekt oder bereits instanziiertes Objekt:
 innerhalb eines Konstruktors ist .prototype nicht zu kodieren
 Prototyping eines Objektes verändert den Umfang der Standard-Methoden und -Eigenschaften zum Objekt
 zur Laufzeit der Scriptmaschine.
 Script-Objekte können nur erweitert werden.
 Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht**
 die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors
 erfolgen und nicht nachträglich per Eigenschaft .prototype .
 Nur für Objekte, die aus einem vordefiniertem Objekt abgeleitet sind, existiert die Eigenschaft .prototype .
 siehe auch .isPrototypeOf() und .hasOwnProperty()

Beispiel für Erweiterung des Script-Objektes Array, das die Eigenschaft .prototype besitzt:

```
function MaximumErmitteln( )
{
    var max = this[0];    // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                        // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}

// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeueArrayMethode = MaximumErmitteln;    // ohne () kodieren !

// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6);    // 6 numerische Elemente

// neue Methode des JScript-Objektes Array aufrufen
alert(Feld.NeueArrayMethode());
```

Beispiel für private Datenstruktur-Objekt mit Methode anhand einer privaten Konstruktor-Methode:
 Es wird die Eigenschaft .prototype **nicht** erzeugt !

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
<!--
function datenstruktur_ausgeben()
{
    with (document)
    {
        write(person.vorname + " " + person.nachname + "<BR>");
        write(person.strasse + " " + person.nummer + "<BR>");
        write(person.plz + " " + person.ort + "<BR>");
        //      Erika Mustermann
        //      Musterstrasse 1
        //      .....
    }

    for (i in person)
    {
        document.write(i + ": " + person[i] + "<BR>");
        //      vorname: Erika
        //      nachname: Mustermann
        //      .....
    }
}

function datenstruktur_erzeugen(vorname, nachname, strasse, nummer, plz, ort, methode)
{
    this.vorname = vorname;
    this.nachname = nachname;
    this.strasse = strasse;
```



```

        this.nummer = nummer;
        this.plz = plz;
        this.ort = ort;
        this.methode = methode;
    }

    person = new datenstruktur_erzeugen
    (
        "Erika",                // Vorname
        "Mustermann",          // Nachname
        "Musterstrasse",        // Strasse
        "1",                    // Nummer
        "10000",                // PLZ
        "Musterstadt",          // Ort
        datenstruktur_ausgeben  // ohne () kodieren
    );

    // alternativ auch kodierbar:
    // var person = {
    //     vorname:"Erika",        // Vorname
    //     nachname:"Mustermann",  // Nachname
    //     strasse:"Musterstrasse", // Strasse
    //     nummer:"1",            // Nummer
    //     plz:"10000",            // PLZ
    //     ort:"Musterstadt"       // Ort
    //     methode:datenstruktur_ausgeben // Ausgabemethode ohne () kodieren
    // };

    // Achtung: Eigenschaft .prototype wird leider nicht (automatisch) erzeugt und ist somit nicht anwendbar !

    alert(person.vorname + "\n" + person.methode); // person.methode ohne () kodieren !
    // -->
</SCRIPT>
</HTML>

```

.pseudoClass Pseudoklasse einer Seite oder @page Regel per Objekt page

":first"	Regel gehört der 1. Seite
":footer"	Regel gehört der Fußnote
":header"	Regel gehört der Kopfnote
":left"	Regel gehört der linken Seite
":left : header"	Regel gehört der Kopfnote Seite links
":left : footer"	Regel gehört der Fußnote Seite links
":right"	Regel gehört der rechten Seite
":right:header"	Regel gehört der Kopfnote Seite rechts
":right:footer"	Regel gehört der Fußnote Seite rechts

Beispiel: @page : left { font-weight:bold;font_style:italic; }

.rating Rating der Media-Datei auf der Timeline

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.wmv"
        STYLE="position:absolute;top:50px;height:100px"
    >
    </t:VIDEO>
    <SPAN ID="ID_Span"
        STYLE="position:absolute;top:165px;"
    >
    </SPAN>
    <BUTTON ID="ID_Button"
        onclick="ID_Span.innerText= ID_Video.rating"
    >
        Klick
    </BUTTON>
</BODY>
</HTML>

```

.readOnly Editierbarkeit eines Text-Controls



Editierung bedeutet Focuserhalt !

false Default
 Objekt ist **nicht** read-only
 also ist es editierbar
 kann also Focus erhalten
 true Objekt ist read-only
 also ist es nicht editierbar
 kann kein also Focus erhalten

.readOnly Art der Implementation des StyleSheets im Dokument ermitteln
 siehe Objekt styleSheet

.readyState aktueller Status des Objektes beim Füllen des Objektes mit Daten
 "uninitialized" Objekt ist nicht initialisiert
 "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung
 "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert
 "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten
 "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert

Beispiel:

alert(document.body.readyState);

.recordNumber Datenquelle-Satznummer eines Datenfeldes

.recordset Zeiger des Standard-Record-Set in einem Datasource-Objekt (DSO)
 Methode . namedRecordset() liefert den Zeiger für Datenquellen-Objekt mit Namen
 also eines beliebige Mitglied im Datasource-Objekt (DSO)

.referrer Url der direkt vorhergehenden Seite der aktuellen Seite
 aktuelle Seite muss durch Link angesprungen sein
 Eingabe in Adresszeile oder per Menü Datei-Öffnen etc. wird nicht verwendet
 ist Leerkette, wenn aktuelle Seite nicht durch Link von der vorherigen aktiviert wurde

.rel Beziehung zwischen Objekt in Quellseite und Nachfolger-Zielseite laut Objekt
 Objekt ist ein Link z.B. A-Tag oder Link-Tag
 nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen
 beim durchforsten der Seiten und Nachfolgerseiten
 vorrangig kodiert in <A> oder <LINK>
 es **muss** zugleich die Eigenschaft .href kodiert und mit **gültigen** Wert belegt sein
 nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)

Beispiel für Webseite mit eigenem Icon ab IE 5.x:

<LINK REL="SHORTCUT ICON" HREF="name.ico">

name.ico: name ist beliebig, auch mit Pfad

ICO-Datei: 32x32 Pixel mit 16 Farben
 oder 16x16 Pixel mit 256 Farben
 ist BMP-Datei, die zu ICO-Datei konvertiert werden muss
 (kann nicht jedes Grafik-Programm, aber Microsoft bietet dafür IconPro an)

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x und NS 6.x):

Seitenteile, die nicht gedruckt werden sollen, in <DIV CLASS="keindruck"> und </DIV>
 oder in und einschliessen.

zwischen <HEAD> und </HEAD> einfügen:

<LINK REL="stylesheet" MEDIA="print" HREF="print.css">

print.css enthält nur

.keindruck {display:none;}

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

Seitenteile, die nur auf Ausdrucken sichtbar sein sollen, in <DIV CLASS="nurdruck"> und </DIV>
 oder in und einschliessen

zwischen <HEAD> und </HEAD> einfügen:

<LINK REL="stylesheet" MEDIA="screen" HREF="screen.css">

screen.css enthält nur

.nurdruck {display:none;}

.repeatCount aktuelle Nummer der Wiederholung bei Loop



siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
function Anzeige()
{
    ID_Div1.innerText = " Aktuelle Wiederholung: "
                        + (ID_Animation.currTimeState.repeatCount + 1);
                        // repeatCount ab 0, Anzeige ab 1
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div1">Aktuelle Wiederholung: 1</DIV>
<DIV ID="ID_Div2"
CLASS="time"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<t:ANIMATEMOTION ID="ID_Animation"
TARGETELEMENT="ID_Div2"
TO="150,0"
BEGIN="0;indefinite"
DUR="1"
AUTOREVERSE="true"
REPEATCOUNT="5"
onrepeat="Anzeige()"
>
</t:ANIMATEMOTION>
<BR>
<BUTTON ID="ID_Button"
onclick= "ID_Div1.innerText='Aktuelle Wiederholung: 1;'"
        + "ID_Animation.beginElement();"
>
Click to restart
</BUTTON>
</BODY>
</HTML>
```

.repeatDur

Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline
verlangt kodierte Eigenschaft .dur oder .repeat
darf nicht kodiert werden mit der Eigenschaft .repeatCount
siehe Objekt currTimeState und Behavior .style.time2

"indefinite" für endlos (Default)

Wert im Time-Format des Behavior

z.B. "h:min:s.f"
"0" entspricht "indefinite"

nicht id.event
nicht id.event+zeit_wert_als_string

Beispiele:

"25:45:10"	25 Stunden, 45 Minuten, 10 Sekunden
"45:35"	45 Minuten, 35 Sekunden
"45:00.275"	45 Minuten, 0,275 Sekunde
"10.5"	10,5 Sekunden

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SEQ ID="ID_Seq"
REPEATDUR="27"
BEGIN="0"
>
<DIV ID="ID_Div1"
CLASS="time_line_klasse"
```



```

        DUR="3"
    >
        3 Sekunden lang anzeigen
    </DIV>
    <DIV    ID="ID_Div2"
        CLASS="time_line_klasse"
        DUR="4"
    >
        4 Sekunden lang anzeigen
    </DIV>
</t:SEQ>
</BODY>
</HTML>

```

.restart generelle Restartmöglichkeit eines Elementes auf der Timeline
 ersetzt die Eigenschaft eventRestart, da diese deprecated ist und nicht mehr verwendet werden darf !!
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:PAR    ID="ID_Par"
        BGIN="indefinite"
        DUR="20"
    >
        <DIV    ID="ID_Div1"
            CLASS="time_line_klasse"
            STYLE="...."
            END="6"
            RESTART="never"
        >
        </DIV>
        <DIV    ID="ID_Div2"
            CLASS="time_line_klasse"
            STYLE="...."
            BEGIN="2"
            END="8"
            RESTART="always"
        >
        </DIV>
    </t:PAR>
    <BUTTON onclick="ID_Par.beginElement();"
    >
        Start
    </BUTTON>
    <BUTTON onclick="ID_Par.endElement();"
    >
        Stop
    </BUTTON>
    <BUTTON onclick="ID_Div1.beginElement();"
        Restart DIV1
    </BUTTON>
    <BUTTON onclick="ID_Div2.beginElement();"
        Restart DIV2
    </BUTTON>
</BODY>
</HTML>

```

.returnValue Returnwert für den Eventhandler festlegen anstelle von return-Anweisung
 Achtung: Wenn kodiert, so wird eine ebenfalls im Eventhandler kodierte die Anweisung
 return;
 ignoriert
 Objekt event
 true Default. Action des Events wurde ausgeführt
 false Action des Events wurde nicht ausgeführt

.returnValue Returnwert des Dialog-Fensters, das mit Methoden
 .showModalDialog() bzw. .showModelessDialog()
 erzeugt wurde



wird gefüllt durch .showModalDialog() bzw. .showModelessDialog()
siehe Objekt window

.rev Beziehung zwischen Objekt in Quellseite und Vorgänger-Zielseite laut Objekt
Objekt ist ein Link z.B. A-Tag oder Link-Tag
nur wichtig für Suchmaschine: dient als Steuerungshinweis für Suchmaschinen
beim durchforsten der Seiten und Vorgängerseiten
vorrangig kodiert in <A> oder <LINK>
es **muss** zugleich die Eigenschaft .href kodiert und mit **gültigen** Wert belegt sein
nur für einen Anker ist zugleich Eigenschaft .rev kodierbar (falls sinnvoll)

.right rechte Pixelposition des Rechteckes um ein Objekt
auch textrectangle Objekt

Beispiel für Nicht-TextRectangle-Objekt:

```
<SCRIPT>
function Anzeigen(ObjektZeiger)
{
    var Rechteck = ObjektZeiger.getBoundingClientRect();

    alert(      "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
              + "\n"
              + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
            );
}
</SCRIPT>
<P onclick="Anzeigen(this)">
```

Beispiel für TextRectangleObjekt:

```
<SCRIPT>
function Anzeigen(ObjektZeiger)
{
    var TextRectangleCollection = document.body.ObjektZeiger.TextRange.getClientRects();

    // .getClientRects() liefert immer Collection der textrectangle Objekte !!!

    var Rechteck = TextRectangleCollection[0];

    alert(      "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
              + "\n"
              + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
            );
}
</SCRIPT>
<BODY>
<DIV onclick="Anzeigen(this)">
    Test
</DIV>
</BODY>
```

.rightMargin Right Margin in Pixel des Dokumentes
Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objekthinhaltes zum Aussenrand
document.body.rightMargin

.RootFolder Root des Laufwerkes liefern, also mit "\\\"
Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_Root = Laufwerk.RootFolder;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_Root);
```

.rowIndex Index der Tabellenzeile in der Collection table.rows
siehe Objekt table.tr

.rows Höhe aller Frames eines Frameset (Höhe des Frameset als Container)

Beispiel:

```
<FRAMESET ROWS="40%, 60%"> // Summe muss immer 100% sein
<FRAMESET ROWS="50, *, 80%"> // 50 Pixel, 80 Pixel und restliche Fensterhöhe
```



.rows	Anzahl der sichtbaren Zeilen der TEXTAREA siehe textarea Objekt
.rowSpan	Anzahl der Zeilen einer Zelle in einer Tabelle siehe Objekt table.tr.td siehe Objekt table.tr.th
.rules	Art der sichtbaren inneren Rahmen einer Tabelle Art der sichtbaren Rahmen zwischen den Tabellenelementen Art des sichtbaren Rahmens um Tabelle siehe Objekt table "all" Rahmen um alle Zeilen und Spalten "cols" Trennlinie zwischen allen Spalten "groups" horizontale Trennlinie zwischen allen THEAD, TBODY's und TFOOT vertikale Trennlinie zwischen allen COLGROUP "none" keine Rahmen und Trennlinien zwischen Tabellenelementen "rows" Trennlinie zwischen allen Zeilen "" keine Rahmen generell (auch nicht um Tabelle)

Beispiel:

```
<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
<THEAD>
  <TR>
    <TD>Kopf Zelle 1</TD>
    <TD>Kopf Zelle 2</TD>
  </TR>
</THEAD>
<TBODY>
  <TR>
    <TD>Body Zeile 1 Zelle 1</TD>
    <TD>Body Zeile 1 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 2 Zelle 1</TD>
    <TD>Body Zeile 2 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 3 Zelle 1</TD>
    <TD>Body Zeile 3 Zelle 2</TD>
  </TR>
</TBODY>
<TFOOT>
  <TR>
    <TD>Fuss Zelle 1</TD>
    <TD>Fuss Zelle 2</TD>
  </TR>
</TFOOT>
</TABLE>
<BUTTON onclick="ID_Tabelle.rules="";">keine Rahmen</BUTTON>
<BUTTON onclick="ID_Tabelle.rules="none";">none</BUTTON>
<BUTTON onclick="ID_Tabelle.rules="all";">all</BUTTON>
<BUTTON onclick="ID_Tabelle.rules="cols";">cols</BUTTON>
<BUTTON onclick="ID_Tabelle.rules="groups";">groups</BUTTON>
<BUTTON onclick="ID_Tabelle.rules="rows";">rows</BUTTON>
```

.scheme	Schema der Interpretation des Wertes laut Eigenschaft .content per meta Objekt z.B. von Datum und Zeit oder Tag-Content-Beschreibung
---------	--

Beispiele:

```
<META scheme="USA" name="date" content="04-05-1962">
<META scheme="Europe" name="date" content="05-04-1962">
```

Tag ID beschreiben

```
<META scheme="customer" name="id" content="test">
```

.scope	Gruppierung, in der eine Zelle liegt siehe Objekt table.tr.td siehe Objekt table.tr.th "row" Zelle enthält Header-Informationen zur Zeile "col" Zelle enthält Header-Informationen zur Spalte "rowgroup" wie "row" aber zur Gruppe aus Zeilen "colgroup" wie "col" aber zur Gruppe aus Spalten
--------	--

.scopeName	Namensraum laut XMLNS-Attribut
------------	--------------------------------



Beispiel:

```
<HTML XMLNS:InetSDK='http://www.test.de'>
<STYLE>
    @media all {InetSDK\;HalloDu { behavior:url (simple.htc) }}
</STYLE>
<SCRIPT>
    function window.onload()          // überschreibt Standard window.onload-Routien !!!!
    {
        // Statuszeile als Ausgabebereich nutzen
        window.status = 'scopeName = ' + Hallo.scopeName + ' ';
        + ' tagUrn = ' + Hallo.tagUrn;
    }
</SCRIPT>
<BODY>
    <InetSDK:HelloWorld ID=Hallo></InetSDK:HalloDu>
</BODY>
</HTML>
```

.screen	Zeiger auf das Objekt screen siehe Objekt window
.screenLeft	X-Koordinate der linken oberen Fensterecke (ohne Bar, Menü, Scrollbalken, Statuszeile etc) bezüglich linke obere Ecke (0,0) vom Bildschirm siehe Objekt window
.screenTop	Y-Koordinate der linken oberen Fensterecke (ohne Bar, Menü, Scrollbalken, Statuszeile etc) bezüglich linke obere Ecke (0,0) vom Bildschirm siehe Objekt window
.screenX	X-Koordinate Maus relativ zum Bildschirm Objekt event
.screenY	Y-Koordinate Maus relativ zum Bildschirm Objekt event
.scroll	Scrollbar-Anzeige ein/aus vor IE 6.x nur BODY-Objekt ab IE 6.x alle HTML-Elemente wenn DOCTYPE im Kopf des Dokumentes kodiert "yes" Default sichtbar "no" nicht sichtbar
.scrollAmount	Scrollschrittweite in Pixel des marquee Objektes Standard ist 6
.scrollDelay	Wartezeit in Millisekunden und laut Eigenschaft .trueSpeed zwischen zwei Scrollschritten eines marquee Objektes Scrollgeschwindigkeit, Fließgeschwindigkeit in Millisekunden wenn .trueSpeed auf false (Standard) , so wenn Wert < 60 so 60 verwendet = 1 Sekunde wenn Wert > 59 so Wert verwendet in Millisekunden wenn .trueSpeed auf true, so Wartezeit = Wert in Millisekunden Standard ist 85 Millisekunden wenn .trueSpeed auf false (Standard) , so Wartezeit = 85 Millisekunden da > 60 wenn .trueSpeed auf true, so Wartezeit = 85 Millisekunden Belegung von .trueSpeed ist egal
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes

Beispiel:

```
<DIV ID="ID_Div" STYLE="overflow:scroll; height=100; width=250; text-align:left">
</DIV>
<INPUT TYPE=button VALUE="anzeigen"
onclick="javascript:alert('Scrollhöhe = ' + ID_Div.scrollHeight);"
>
```

.scrolling	Scrollenbar erzeugen "auto" Default automatisch Scrollbar erzeugt, wenn nötig "no" nie Scrollbar erzeugen, also kein Scrollen "yes" immer Scrollbar erzeugen, also Scrollen
------------	---



.scrollLeft Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes
 nur nutzbar nach dem kompletten Laden des Dokumentes
 immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind

Beispiel:

```
<DIV STYLE="position:absolute; width:200px; height:100px; overflow:scroll"
  onclick="alert(this.scrollLeft);"
>
  <SPAN STYLE="width:250px"> . . . </SPAN>
</DIV>
```

.scrollTop Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes
 nur nutzbar nach dem kompletten Laden des Dokumentes
 immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind

Beispiel:

```
<DIV STYLE="position:absolute; width:200px; height:100px; overflow:scroll"
  onclick="alert(this.scrollTop)"
>
  <SPAN STYLE="width:250px"> . . . </SPAN>
</DIV>
```

.scrollWidth Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes

Beispiel:

```
<SCRIPT>
function Anzeige()
{
    var ScrollBreite = ID_Div.scrollWidth;
    var OffsetBreite = ID_Div.offsetWidth;
    var Differenz = iScrollWidth - iOffsetWidth;

    alert(    "OffsetWidth = " + OffsetBreite
            + "\nScrollWidth = " + ScrollBreite
            + "\nDifferenz = " + Differenz
            );
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="overflow:scroll;height=200;width=250;text-align:left">
</DIV>
<INPUT TYPE=button VALUE="anzeigen" onclick="Anzeige()">
```

.search Teil des Wertes des Eigenschaft .href
 Teil folgt direkt dem Fragezeichen
 wird als Querystring oder Formdata bezeichnet
 hat nichts mit der Suche auf Webseiten zu tun
 Hinweis: Fragezeichen-Anhang an der HREF dient zur Übergabe von String-Werten einer Webseite an eine andere:
 Quellseite besitzt HREF mit " ...?....."
 ruft Zielseite mit diesem HREF auf
 Zielseite wurde von Quellseite aufgerufen
 liest den Teil von HREF hinter dem ? als Zeichenkettendaten

Beispiel:

```
document.location.search;
```

.sectionRowIndex Index der Tabellenzeile in der Collection
 table.tBody.rows
 bzw. table.tFoot.rows
 bzw. table.tHead.rows
 Zeile muss im **existierenden** Tabellenteil TOBODY bzw. TFOOT bzw. THEAD liegen
 siehe Objekt table.tr

SECURITY temporäre Sicherheit des IFRAME
 wird an Kinder weitervererbt
 ab IE 6.0
 "restricted"
 entspricht als hätte der User im Browser-Menü "Extra-Internet Optionen-Sicherheit" den Eintrag "Programme und Dateien in einem IFRAME starten" auf "Deaktivieren" gesetzt (falls nicht durch User bereits getan)
 bewirkt somit die Sperrung der Ausführung von Scripten wie javascript, vbscript **und** Protokollen in URLs (z.B. http), die im Source-File kodiert sind und im IFRAME aktiviert werden sollen
 Hinweis: wenn restricted, so alle untergeordneten IFRAME ebenfalls restricted

Beispiel 1:

wenn restricted, so nachfolgender Script-Code im FRAME und IFRAME nicht ausgeführt:




```

<IFRAME SECURITY="restricted">
  <A HREF="javascript:alert('funktioniert nicht bei FRAME oder IFRAME!');">
    JavaScript Link
  </A>
</IFRAME>

```

sondern für den Link ein neues Fenster geöffnet

Beispiel 2:

wenn restricted, so nachfolgender Script-Code im IFRAME nicht ausgeführt:

```

<IFRAME SECURITY="restricted" SRC="http://www.microsoft.com"></IFRAME>

```

sondern für den Link ein neues Fenster geöffnet

.segmentDur

Dauer der Timeline der Wiederholungen laut autoReverse
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
  function ZeitHolen()
  {
    ID_Span3.innerText += ID_Span1.currTimeState.segmentDur;
    ID_Span4.innerText += ID_Span2.currTimeState.segmentDur;
  }
</SCRIPT>
</HEAD>
<BODY>
  <t:EXCL onbegin="ZeitHolen();">
    <t:PRIORITYCLASS PEERS="pause">
      <SPAN ID=" ID_Span1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="10"
      >
        <H3>Animation 1</H3>
        <P>im Wechsel mit Animation 2</P>
      </SPAN>
      <SPAN ID=" ID_Span2"
        CLASS="time_line_klasse"
        BEGIN="5"
        DUR="5"
      >
        <H3>Animation 2</H3>
        <P>im Wechsel mit Animation 1</P>
      </SPAN>
    </t:PRIORITYCLASS>
  </t:EXCL>
  <BR>
  <SPAN ID=" ID_Span3"></SPAN>&nbsp;&nbsp;&nbsp;Sekunden
  <SPAN ID=" ID_Span4"></SPAN>&nbsp;&nbsp;&nbsp;Sekunden
</BODY>
</HTML>

```

.segmentTime

aktueller Zeitpunkt in der Timeline der Wiederholungen laut autoReverse
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
  function WindowOnLoad()
  {
    // Timer mit Intervall von 100 Millisekunden
    window.setInterval(Anzeigen, 100);
  }

  function Anzeigen()
  {

```



```

        ID_Div2.innerHTML = "simpleTime:&nbsp;"
                          + (ID_Animation.currTimeState.simpleTime);

        ID_Div3.innerHTML = "segmentTime:&nbsp;"
                          + (ID_Animation.currTimeState.segmentTime);

        ID_Div4.innerHTML = "activeTime:&nbsp;"
                          + (ID_Animation.currTimeState.activeTime);
    }

    function Wiederholen()
    {
        ID_Div1.innerHTML = "repeatCount:&nbsp;"
                          + (ID_Animation.currTimeState.repeatCount + 1);
                          // repeatCount ab 0, aber Anzeige ab 1
    }
    window.onload = WindowOnLoad;    // mit () so sofort ausgeführt
                                      // ohne () so reiner Zeiger

</SCRIPT>
</HEAD>
<BODY>
    <DIV>
        <DIV ID="ID_Div1">repeatCount:&nbsp;1</DIV>
        <DIV ID="ID_Div2">simpleTime:&nbsp;0</DIV>
        <DIV ID="ID_Div3">segmentTime:&nbsp;0</DIV>
        <DIV ID="ID_Div4">activeTime:&nbsp;0</DIV>
    </DIV>

    <DIV ID="ID_DivTimeLine"
        CLASS="time_line_klasse"
        STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;" >
    </DIV>
    <t:ANIMATEMOTION          ID="ID_Animation"
                              TARGETELEMENT="ID_DivTimeLine"
                              TO="340,40"
                              DUR="2"
                              AUTOREVERSE="true"
                              REPEATCOUNT="5"
                              FILL="freeze"
                              onrepeat="Wiederholen()"
    >
    </t:ANIMATEMOTION>
</BODY>
</HTML>

```

SEGMENTTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.selected	<p>Selektionsstatus der Option, also Objektes document.select.option des Objektes document.select standardgemäß ist immer die oberste Option selektiert es kann aus einer Optionengruppe immer nur genau 1 Option selektiert sein</p> <p>true Option ist selektiert false Default Option ist nicht selektiert</p>
.selectedIndex	<p>Index der Option in der Collection document.select.options des Objektes select jedes option Objekt eines select Objektes wird im Feld, das Teil des select Objektes ist, referenziert es kann immer nur eine Option innerhalb einer Optionsgruppe selektiert sein</p>
.selector	Seiten-Selektor per Objekt page
.self	Referenz auf das aktuelle Fenster oder den aktuellen Frame
.self	<p>Zeiger auf das aktuelle Fenster innerhalb der Fensterhierarchie im FRAME</p> <p>ist synonym zu window siehe Objekt window</p>
.SerialNumber	<p>Seriennummer des Laufwerkes liefern</p> <p>Achtung: Die Seriennummer lässt eine eindeutige Identifizierung der Festplattenhardware zu vorallem dann, wenn zusätzlich Userdaten zum PC bekannt sind.</p>

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
```



```

var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk              = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName   = Laufwerk.VolumeName;
var Laufwerk_SerienNummer = Laufwerk.SerialNumber;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " "
      + Laufwerk_SerienNummer);

```

.shape Form/Gestalt eines Objektes
 siehe auch Eigenschaft .coords
 "circ" oder "circle" Kreisform
 "poly" oder "polygon" Vieleck (Polygon)
 "rect" oder "rectangle" Viereck

.ShareName öffentlicher Netzwerkname des Laufwerkes liefern

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_ShareName = Laufwerk.ShareName;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_ShareName);

```

.shiftKey SHIFT-Tasten-Status
 Objekt event
 false SHIFT-Taste ist nicht gedrückt
 true SHIFT-Taste ist gedrückt

Beispiel:

```

<HEAD>
<SCRIPT>
function InnerTextSetzen(Zeiger, Kette)
{Zeiger.innerText=Kette;}

function init()
{
    InnerTextSetzen(ID_Span1,'false');
    InnerTextSetzen(ID_Span2,'false');
}
function ShiftDown()
{
    if (event.shiftLeft)
    { InnerTextSetzen(ID_Span1,'true'); }
    else
    {
        if (event.shiftKey)
        { InnerTextSetzen(ID_Span2,'true'); }
    }
}

function ShiftUp()
{
    if (!event.shiftKey)
    {
        InnerTextSetzen(ID_Span1,'false');
        InnerTextSetzen(ID_Span2,'false');
    }
}
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="ShiftDown();" onkeyup="ShiftUp();">
    <TABLE>
        <TR>
            <TD><I>Linke Shift-Taste gedrueckt</I></TD>
            <TD><I>Recchte Shift-Taste gedrueckt</I></TD>
        </TR>
        <TR>
            <TD ALIGN="center">
                <SPAN ID="ID_Span1"></SPAN>
            </TD>
            <TD ALIGN="center">
                <SPAN ID="ID_Span2"></SPAN>
            </TD>
        </TR>
    </TABLE>
</BODY>

```



.shiftLeft SHIFT-Taste links Status
 nur für Dokument mit Focus
 Objekt event
 false linke SHIFT-Taste ist nicht gedrückt
 true linke SHIFT-Taste ist gedrückt

Beispiel:

```
<HEAD>
<SCRIPT>
function InnerTextSetzen(Zeiger, Kette)
{Zeiger.innerText=Kette;}

function init()
{
    InnerTextSetzen(ID_Span1,'false');
    InnerTextSetzen(ID_Span2,'false');
}
function ShiftDown()
{
    if (event.shiftLeft)
    { InnerTextSetzen(ID_Span1,'true'); }
    else
    {
        if (event.shiftKey)
        { InnerTextSetzen(ID_Span2,'true'); }
    }
}

function ShiftUp()
{
    if (!event.shiftKey)
    {
        InnerTextSetzen(ID_Span1,'false');
        InnerTextSetzen(ID_Span2,'false');
    }
}
</SCRIPT>
</HEAD>
<BODY onload="document.body.focus(); init()" onkeydown="ShiftDown();" onkeyup="ShiftUp();">
    <TABLE>
        <TR>
            <TD><I>Linke Shift-Taste gedrueckt</I></TD>
            <TD><I>Rechte Shift-Taste gedrueckt</I></TD>
        </TR>
        <TR>
            <TD ALIGN="center">
                <SPAN ID="ID_Span1"></SPAN>
            </TD>
            <TD ALIGN="center">
                <SPAN ID="ID_Span2"></SPAN>
            </TD>
        </TR>
    </TABLE>
</BODY>
```

.ShortName Ordnerbezeichner als 8.3 Bezeichner liefern

Beispiel:

```
var OrdnerName                = "c:\\windows\\desktop\\";
var DateiSystem              = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                    = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ShortName);
```

.ShortName Dateibezeichner als 8.3 Bezeichner liefern

Beispiel:

```
var DateinameMitPfad         = "c:\\test.txt";
var DateiSystem              = new ActiveXObject("Scripting.FileSystemObject");
var Datei                    = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.ShortName);
```

.ShortPath Pfad des Ordners als 8.3 Pfad liefern

Beispiel:

```
var OrdnerName                = "c:\\windows\\desktop\\";
var DateiSystem              = new ActiveXObject("Scripting.FileSystemObject");
```



```
var Ordner                = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.ShortPath);
```

.ShortPath Pfad der Datei als 8.3 Pfad liefern

Beispiel:

```
var DateinameMitPfad      = "c:\\test.txt";
var DateiSystem           = new ActiveXObject("Scripting.FileSystemObject");
var Datei                 = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.ShortPath);
```

.simpleDur Dauer einer Wiederholung
Wiederholung erzeugt per Eigenschaft .autoReverse auf true setzen
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
function Anzeige1()
{
    ID_Div1.innerText = " Anzahl Wiederholungen: "
                      + (ID_Animation.currTimeState.repeatCount + 1);
                      // repeatCount ab 0, Anzeige ab 1
}

function Anzeige2()
{
    ID_Span.innerHTML = " Dauer jeder Wiederholung: "
                      + (ID_Animation.currTimeState.simpleDur)
                      + ' Sekunden';
}
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Div0"
CLASS="time_line_klasse"
DUR=".01"
REPEATCOUNT="indefinite"
FILL="hold"
onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
>
0
</SPAN>
<BR>
<DIV ID="ID_Div1"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
Anzahl Wiederholungen: 1
</DIV>
<SPAN ID="ID_Span">Dauer jeder Wiederholung: </SPAN>
<DIV ID="ID_Div2"
CLASS="time_line_klasse"></DIV>
<t:ANIMATEMOTION ID="ID_Animation"
TARGETELEMENT=" ID_Div1"
TO="325,0"
BEGIN="0; indefinite;"
DUR="2"
AUTOREVERSE="true"
REPEATCOUNT="5"
onrepeat="Anzeige1()"
>
</t:ANIMATEMOTION>
<BR>
<BUTTON ID="ID_Button1" onclick="Anzeige2()">
Anzeige Dauer
</BUTTON>
<BUTTON ID="ID_Button2" onclick=" ID_Animation.beginElement();">
Click to restart
</BUTTON>
</BODY>
</HTML>
```



.simpleTime aktueller Punkt auf Timeline
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT LANGUAGE="JScript">
function WindowOnLoad()
{
// Timer mit Intervall von 100 Millisekunden
window.setInterval(Anzeigen, 100);
}

function Anzeigen()
{
ID_Div2.innerHTML = "simpleTime:&nbsp;"
+ (ID_Animation.currTimeState.simpleTime);

ID_Div3.innerHTML = "segmentTime:&nbsp;"
+ (ID_Animation.currTimeState.segmentTime);

ID_Div4.innerHTML = "activeTime:&nbsp;"
+ (ID_Animation.currTimeState.activeTime);
}

function Wiederholen()
{
ID_Div1.innerHTML = "repeatCount:&nbsp;"
+ (ID_Animation.currTimeState.repeatCount + 1);
// repeatCount ab 0, aber Anzeige ab 1
}
window.onload = WindowOnLoad; // mit () so sofort ausgeführt
// ohne () so reiner Zeiger

</SCRIPT>
</HEAD>
<BODY>
<DIV>
<DIV ID="ID_Div1">repeatCount:&nbsp;1</DIV>
<DIV ID="ID_Div2">simpleTime:&nbsp;0</DIV>
<DIV ID="ID_Div3">segmentTime:&nbsp;0</DIV>
<DIV ID="ID_Div4">activeTime:&nbsp;0</DIV>
</DIV>

<DIV ID="ID_DivTimeLine"
CLASS="time_line_klasse"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;" >
</DIV>
<t:ANIMATEMOTION ID="ID_Animation"
TARGETELEMENT="ID_DivTimeLine"
TO="340,40"
DUR="2"
AUTOREVERSE="true"
REPEATCOUNT="5"
FILL="freeze"
onrepeat="Wiederholen()"
>
</t:ANIMATEMOTION>
</BODY>
</HTML>
```

.size Fontgröße
 1 bis 7
 1 kleinste Größe
 7 größte Größe

.size Dimension eines Input-Objektes in Zeichenanzahl

Beispiel:

```
<INPUT TYPE=text SIZE=33>
```



.size Anzahl der Zeilen in einer List-Box als select Objekt

.Size Ordnergrösse in Bytes liefern

Beispiel:

```
var OrdnerName      = "c:\\windows\\desktop\\";
var DateiSystem     = new ActiveXObject("Scripting.FileSystemObject");
var Ordner          = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Size);
```

.Size Dateigrösse in Bytes liefern

Beispiel:

```
var DateinameMitPfad = "c:\\test.txt";
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Datei            = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.Size);
```

.sourceIndex Index des Objektes in der Collection document.all

Beispiel:

```
<SCRIPT>
function Handler()
{
    // Element referenzieren das Event verarbeitet
    var ElementZeiger = event.srcElement;

    var Index= ElementZeiger.sourceIndex;
    var TagName= ElementZeiger.tagName;

    if(TagName=="!"){TagName="COMMENT";}

    ID_TD1.innerHTML=Index;
    ID_TD2.innerHTML=TagName;

    if (Index-1>0)
    {
        TagName=document.all[Index-1].tagName;

        if(TagName=="!"){TagName="COMMENT";}

        ID_TD3.innerHTML=sTagName;
    }
    else
    { ID_TD3.innerHTML="nicht vorhanden"; }

    if (Index+1<document.all.length)
    {
        TagName=document.all[Index+1].tagName;

        if(TagName=="!"){TagName="COMMENT";}

        ID_TD4.innerHTML=TagName;
    }
    else
    { ID_TD4.innerHTML="nicht vorhanden"; }
}
</SCRIPT>
<BODY onmousemove="Handler()">
<TABLE>
<TR><TD>Source Index:</TD><TD ID="ID_TD1"></TD></TR>
<TR><TD>Objekt-Name:</TD><TD ID="ID_TD2"></TD></TR>
<TR><TD>Vorhergehendes Objekt:</TD><TD ID="ID_TD3"></TD></TR>
<TR><TD>Naechstes Objekt:</TD><TD ID="ID_TD4"></TD></TR>
</TABLE>
</BODY>
```

.sourceURL Url der Media-Datei des MediaItem Objektes
siehe Behavior .style.mediaBar

.span Anzahl der Spalten innerhalb einer Spaltengruppe einer Tabelle
siehe Objekt table.col und Objekt table.colGroup

Beispiel:

```
<TABLE BORDER="2">
<COLGROUP SPAN="3" STYLE="color:green;background:black">
<COL SPAN="2" STYLE="color:red">
```



</COLGROUP>

....

</TABLE>

.specified

prüfen ob Objekt Attribute hat
 true, so Attribute ist vorhanden
 false, so keine Attribute vorhanden

Beispiel:

```
<SCRIPT>
function Anzeigen()
{
    var FeldAllerAttribute = ID_List.attributes;
    alert(FeldAllerAttribute(0).nodeName);    // Knotenname

    for( var i=0; i< FeldAllerAttribute.length; i++)
    {
        // neues LI-Element als Knoten der Liste anhängen
        var NeuesListenElement = document.createElement("LI");
        ID_List.appendChild(NeuesListenElement);

        // Wert des neuen LI-Elementes ist Text, also Textknoten
        var WertNeuesListenElement = document.createTextNode(
            i
            + " "
            + FeldAllerAttribute(i).nodeName
            + " = "
            + FeldAllerAttribute(i).nodeValue
        );
        NeuesListenElement.appendChild(WertNeuesListenElement);

        // auf specified prüfen
        if(FeldAllerAttribute(i).nodeValue != null )
        {
            alert(    FeldAllerAttribute(i).nodeName
                    + " specified: "
                    + FeldAllerAttribute(i).specified    // boolean
            );
        }
    }
}
</SCRIPT>
<UL ID="ID_List" onclick = " Anzeigen()">
    <LI>Klick mich
</UL>
```

.speed

aktuelle Wiedergabegeschwindigkeit des Kindes (Run time speed)
 Beispiel: Eltern mit 50 % Geschwindigkeit
 Kind mit 50 % Geschwindigkeit der Eltern
 also aktuelle Wiedergabegeschwindigkeit = 25 %
 siehe Objekt currTimeState und Behavior .style.time2
 Floating-point
 in Prozent

Beispiel 1:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:PAR SPEED="2" >
        <t:KIND ID="ID_Kind"
            BEGIN="1"
            SRC="http://www.test.de/media/test.wmv"
            SPEED="1.5"
        >
    </t:KIND>
    </t:PAR>
    <BUTTON ID="ID_Button1"
        CLASS="time_line_klasse"
        REPEATCOUNT="indefinite"
        onclick="alert( 'Run time speed des Kindes: ' + ID_Kind.currTimeState.speed );"
    >
```



```

        Run time speed des Kindes
    </BUTTON>
    <BUTTON
        ID="ID_Button2"
        CLASS="time_line_klasse"
        REPEATCOUNT="indefinite"
        onclick="alert( 'SPEED-Attribut des Kindes: ' + ID_Kind.speed );"
    >
        SPEED-Attribut des Kindes
    </BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<SCRIPT>
    function Aendern()
    {
        ID_Video.updateMode = ID_Select1.options.value;
        ID_Video.speed      = ID_Select1.options.value;
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:VIDEO
        ID="ID_Video"
        SRC="test.avi"
        UPDATEMODE="reset"
        STYLE="width:175px; height:150px;"
    >
    </t:VIDEO>
    <BR>
    updateMode waehlen:
    <SELECT NAME="ID_Select1">
        <OPTION VALUE="auto">Auto</OPTION>
        <OPTION VALUE="reset" SELECTED>Reset</OPTION>
    </SELECT>
    <BR>
    Geschwindigkeit waehlen:
    <SELECT NAME="ID_Select2">
        <OPTION VALUE="0.25">25%</OPTION>
        <OPTION VALUE="0.50">50%</OPTION>
        <OPTION VALUE="0.75">75%</OPTION>
        <OPTION VALUE="1" SELECTED>100% </OPTION>
        <OPTION VALUE="2" SELECTED>200% </OPTION>
    </SELECT>
    <BR>
    <BUTTON ID="ID_Button1" onClick="Aendern();">
        Geschwindigkeit aendern
    </BUTTON>
    <BUTTON ID="ID_Button1" onClick="document.body.beginElement();">
        Restart
    </BUTTON>
</CENTER>
</BODY>
</HTML>

```

.SQRT1_2 Quadratwurzel von 0,5 also von 1 dividiert durch Quadratwurzel von 2
siehe Script-Objekt Math

.SQRT2 Quadratwurzel von 2
siehe Script-Objekt Math

.src Url der Daten z.B. vom Image in normaler Auflösung

Hinweis: Eine Kodierung der Url ohne Einschluss in " " bzw. ' ' kann in der Regel nur erfolgen, wenn die Url bereits ein Wert vom Format der Eigenschaft .src ist. Empfehlung: Per new ein Objekt erzeugen und dann .src belegen.
.src bzw. das SRC-Attribut kann mit einer Url belegt werden, die innerhalb " " bzw. ' ' kodiert wurde. Alternativ ist der Wert der Eigenschaft .src eines anderen Objektes zuweisbar, aber ohne Kodierung von " " bzw. ' '.



Beispiel:

```
var Pfad='./test/';
var BildName = 'testbild.jpg';
var BildObjekt1 = new Image(10,20);
BildObjekt1.src= "" + Pfad + BildName + "";
BildObjekt1.src= Pfad + BildName; // könnte Fehler erzeugen bzw. nicht ausgeführt werden
var BildObjekt2 = new Image(100,200);
BildObjekt2.src = BildObjekt1.src;
BildObjekt2.src = "" + BildObjekt1.src + ""; // könnte Fehler erzeugen bzw. nicht ausgeführt werden
```

Beispiel:

```
<BGSOUND SRC="file:///c:/test/wav/test.wav">
```

.src Url einer Media-Datei auf der Timeline
 nur lesen beim Objekt laut `object.playList.item()` bzw. `object.playList.aktiveTrack`

.srcElement Referenz auf Objekt das das Event auslöst
 Objekt event

Beispiel 1:

```
<SCRIPT LANGUAGE="JScript">
function selectWord()
{
    var ZeigerAufObjekt = window.event.srcElement ;

    if (!ZeigerAufObjekt.isTextEdit)
    { ZeigerAufObjekt = window.event.srcElement.parentTextEdit; }

    if (ZeigerAufObjekt != null)
    {
        var TextBereich = ZeigerAufObjekt.createTextRange();
        TextBereich.moveToElementText(window.event.srcElement);
        TextBereich.collapse();
        TextBereich.expand("word");
        TextBereich.select();
    }
}
</SCRIPT>
```

Beispiel 2:

```
<STYLE>
.normal { background-color: "#FFFFFF"; color: "#000000"; font-weight: normal; font-size: 8pt; font-family: Arial; }
.accessible { background-color: "beige"; font-weight: bold; font-size: 10pt; }
</STYLE>
<SCRIPT>
function StyleWechsel()
{
    // Input-Objekt
    event.srcElement.className="accessible"; // Input-Objekt

    // Label-Objekt
    var ZeigerAuf Label =eval(event.srcElement.id + "_Label ");
    // ID ist "ID_Input"
    // also "ID_Input" + "_Label" = "ID_Input_Label"
    // Zeichenkettenoperation leider nötig, wenn mehrere
    // Objekte mit Eventerzeugung vorhanden wären
    // und ebenfalls nötig wegen Bezug im Label auf das
    // ID des Objektes, das Label haben soll
    ZeigerAuf Label.className="accessible";
}
</SCRIPT>
<LABEL FOR="ID_Input" oInput" CLASS="normal" ID="ID_Input_Label">Text eingeben</LABEL>
<INPUT TYPE="text" CLASS="normal" onfocus="StyleWechsel()" ID="ID_Input">
```

Beispiel 3:

```
<HTML>
<HEAD>
<SCRIPT>
function HandlerFuerOnMoveStart()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}
```



```

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandlerFuerOnMoveEnd()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

// 2-D Positionierung einschalten
document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offserTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

.standby Zeichenkette für Standby-Funktionalität des per OBJECT-Tag eingebundenen Objektes

.start Start eines Videoclips oder VRML-Datei

.state Status der Timeline – Variante 1
 siehe Objekt currTimeState und Behavior .style.time2
 0 Timeline inaktiv
 1 Timeline aktiv
 2 Timeline verarbeitet media file
 Element ist aktiv
 nur für Element das media file verarbeitet
 3 Timeline sucht einen Punkt im media file
 Element ist aktiv
 nur für Element das media file verarbeitet
 4 Timeline ist holding für das aktuelle Element und Element wartet auf Ende der Timeline
 der Eltern
 Element ist nicht aktiv
 Hinweis zu media file: Audio, Video
 zu Attribut FILL: wenn "hold" oder freeze" dann wartet Element auf
 die Synchronisation mit der
 Timeline des Elternobjektes
 wenn "hold" Element kann keine Events verarbeiten
 ist inaktiv
 wenn "freeze" Element ist aktiv

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        switch (ID_Video.currTimeState.state)
        {
            case 1:

```



```

        if (ID_Video.currTimeState.isPaused == true)
        {
            ID_Button2.disabled = true;
            ID_Button3.disabled = true;
            ID_Button4.disabled = false;
            ID_Button5.disabled = false;
        }
        else
        {
            ID_Button2.disabled = true;
            ID_Button3.disabled = false;
            ID_Button4.disabled = true;
            ID_Button5.disabled = false;
        }
        break;
    case 0:
        ID_Button2.disabled = false;
        ID_Button3.disabled = true;
        ID_Button4.disabled = true;
        ID_Button5.disabled = true;
        break;
    case 4:
        ID_Button2.disabled = false;
        ID_Button3.disabled = true;
        ID_Button4.disabled = true;
        ID_Button5.disabled = true;
        break;
    }
}
</SCRIPT>
<SCRIPT FOR="document" EVENT="onclick" LANGUAGE="JScript">
    Anzeige();
</SCRIPT>
</HEAD>
<BODY onload=" Anzeige()">
    <t:VIDEO ID="ID_Video"
        CLASS="time_line_klasse"
        BEGIN="indefinite"
        STYLE="position:absolute;top:90px;height:150px;"
        FILL="remove"
        SRC="/test /media/movie.avi"
    >
</ t:VIDEO>
<SPAN ID="ID_Span"
    STYLE="position:absolute;top:255px;"
>
    Status: 0
</SPAN>
<P STYLE="position:absolute;top:280px;">
    <BUTTON ID="ID_Button1"
        onclick=
            "ID_Span.innerText='Status: ' + ID_Video.currTimeState.state"
    >
        Aktueller Status
    </BUTTON>
    <BUTTON ID="ID_Button2"
        onclick="ID_Video.beginElement();"
    >
        Beginn
    </BUTTON>
    <BUTTON ID="ID_Button3"
        onclick="ID_Video.pauseElement();"
    >
        Pause
    </BUTTON>
    <BUTTON ID="ID_Button4"
        onclick="ID_Video.resumeElement();"
    >
        Resume
    </BUTTON>
    <BUTTON ID="ID_Button5"
        onclick=
            "ID_Video.endElement();ID_Span.innerText='Status: 0'"
    >

```



```

                Stop
            </BUTTON>
        </P>
    </BODY>
</HTML>

```

.stateString Status der Timeline – Variante 2
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    {
        switch (ID_Video.currTimeState.stateString)
        {
            case "active":
                if (ID_Video.currTimeState.isPaused == true)
                {
                    ID_Button2.disabled = true;
                    ID_Button3.disabled = true;
                    ID_Button4.disabled = false;
                    ID_Button5.disabled = false;
                }
                else
                {
                    ID_Button2.disabled = true;
                    ID_Button3.disabled = false;
                    ID_Button4.disabled = true;
                    ID_Button5.disabled = false;
                }
                break;

            case "inactive":
                ID_Button2.disabled = false;
                ID_Button3.disabled = true;
                ID_Button4.disabled = true;
                ID_Button5.disabled = true;
                break;

            case "holding":
                ID_Button2.disabled = false;
                ID_Button3.disabled = true;
                ID_Button4.disabled = true;
                ID_Button5.disabled = true;
                break;
        }
    }
</SCRIPT>
<SCRIPT FOR="document" EVENT="onclick" LANGUAGE="JScript">
    Anzeige();
</SCRIPT>
</HEAD>
<BODY onload="Anzeige()">
    <t:VIDEO ID="ID_Video"
        CLASS="time_line_klasse"
        SRC="test.avi"
        BEGIN="indefinite"
        FILL="remove"
        STYLE="position:absolute;top:90px;height:150px;"
    >
    </ t:VIDEO>
    <SPAN ID="ID_Span"
        STYLE="position:absolute;top:255px;"
    >
        Status: 0
    </SPAN>
    <P STYLE="position:absolute;top:280px;">
        <BUTTON ID="ID_Button1"
            onclick=

```



```

        "ID_Span.innerText='Status: ' + ID_Video.currTimeState.state"
    >
        Aktueller Status
    </BUTTON>
    <BUTTON ID="ID_Button2"
        onclick="ID_Video.beginElement();"
    >
        Beginn
    </BUTTON>
    <BUTTON ID="ID_Button3"
        onclick="ID_Video.pauseElement();"
    >
        Pause
    </BUTTON>
    <BUTTON ID="ID_Button4"
        onclick="ID_Video.resumeElement();"
    >
        Resume
    </BUTTON>
    <BUTTON ID="ID_Button5"
        onclick=
            "ID_Video.endElement();ID_Span.innerText='Status: 0'"
    >
        Stop
    </BUTTON>
</P>
</BODY>
</HTML>

```

.status

Selektionsstatus eines Control-Elementes
 Control-Element: kann durch User interaktiv verändert werden
 wird bei Ceckbox-Control auch verändert durch Eigenschaft **.indeterminate**
 bei Statusveränderung wird Event **onpropertychange** erzeugt

false	Default außer bei textArea Objekt
	Control ist nicht selektiert
true	Control ist selektiert
null	Control ist nicht initialisiert
	Default bei textArea Objekt

Beispiel:

```

<INPUT TYPE=checkbox
        ID="ID_InputCheckbox"
        CHECKED
        DISABLED
    >
    <SPAN STYLE="font-weight:bold"
        onclick="ID_InputCheckbox.status=false"
    >
        ablehnen
    </SPAN>
    <SPAN STYLE="font-weight:bold"
        onclick="ID_InputCheckbox.status=true"
    >
        annehmen
    </SPAN>

```

.status

enthält aktuellen Text der Statuszeile des Fensters (nicht den Standardtext)
 siehe **.defaultStatus**
 Verwendung in Eventhandler-Funktion: Es muss **return true**; kodiert werden.
 siehe Objekt **window**

Beispiel für Text buchstabenweise in Statuszeile anzeigen:

Der aktuelle Statuszeilentext wird als Teilkette von Position 0 bis **zeichen_nr** angezeigt, wobei **zeichen_nr** pro Anzeige um 1 erhöht wird.

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var statuszeile    = new Array();
    statuszeile[0]     = "Text0";
    statuszeile[1]     = "Text1";
    statuszeile[2]     = "Text2";

```




```

var statuszeile_nr = 0;
var zeichen_nr = 0;

function textanzeigen ()
{
    if (position < statuszeile[statuszeile_nr].length) // Länge ab 1
    {
        // nächste Teilkette des aktuellen Statuszeilentextes anzeigen
        window.status = statuszeile[statuszeile_nr].substring(0, zeichen_nr);

        // nächste Position
        position++;
        setTimeout("textanzeigen()", 100);
    }
    else
    {
        // aktuelle Statuszeilentext komplett anzeigen
        window.status = statuszeile[statuszeile_nr];

        // nächste Statuszeile adressieren
        statuszeile_nr++;
        if (statuszeile_nr >= statuszeile.length)
        { statuszeile_nr = 0; }

        // Position 0 einstellen
        zeichen_nr = 0;

        // Start der buchstabenweise Anzeige
        setTimeout("textanzeigen()", 1000);
    }
}

// -->
</SCRIPT>
</HEAD>
<BODY onLoad="textanzeigen()">
</BODY>
</HTML>

```

Beispiel für automatisch wechselnder Text in der Statuszeile:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var statuszeile = new Array();
    statuszeile[0] = "Text0";
    statuszeile[1] = "Text1";
    statuszeile[2] = "Text2";

    var statuszeile_nr = 0;

    function textanzeigen()
    {
        window.status = statuszeile[statuszeile_nr];
        statuszeile_nr++;
        if (statuszeile_nr >= statuszeile.length) { statuszeile_nr = 0; }
        setTimeout("textanzeigen()", 1000);
    }

// -->
</SCRIPT>
</HEAD>
<BODY onLoad="textanzeigen()">
</BODY>
</HTML>

```

Beispiel für dauerhaftes Scrollen eines Textes in der Statuszeile:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext = "";
    var zahler = scrolltext_gesamt.length;

```



```

function scrollen()
{
    if (zahler == scrolltext_gesamt.length)
    {
        zahler = 0;
        scrolltext = scrolltext_gesamt;
    }
    else
    {
        zahler++; // 1 bis scrolltext_gesamt.length
        // Blank vorsetzen, also Kette um 1 Zeichen verlängern
        scrolltext = " "+scrolltext;

        // rausgerutschtes Zeichen abschneiden
        scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
        // Angaben ab 0
    }

    window.status = scrolltext;

    setTimeout("scrollen()", 100);
}
// -->
</SCRIPT>
</HEAD>
<BODY onLoad="scrollen()">
</BODY>
</HTML>

```

Beispiel für einen Scrolltext in der Statusleiste, der angehalten wird, wenn Mauszeiger sich über einen Linkt bewegt:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var scrolltext_gesamt = "Dieser Text scrollt!";
    var scrolltext = "";
    var zahler = scrolltext_gesamt.length;
    var id;

    function scrollen()
    {
        if (zahler >= scrolltext_gesamt.length)
        {
            zahler = 0;
            scrolltext = scrolltext_gesamt;
        }
        else
        {
            zahler++; // 1 bis scrolltext_gesamt.length
            // Blank vorsetzen, also Kette um 1 Zeichen verlängern
            scrolltext = " "+scrolltext;

            // rausgerutschtes Zeichen abschneiden
            scrolltext = scrolltext.substring(0, scrolltext_gesamt.length - 1);
            // Angaben ab 0
        }

        window.status = scrolltext;
        id=setTimeout("scrollen()", 100);
    }

    function scrollen_stoppen()
    {clearTimeout(id);}

// -->
</SCRIPT>
</HEAD>

<BODY onLoad="scrollen()">
    Bewegen Sie den Mauspfel &uuml;ber diesen
    <A      HREF="http://www.test.de"
        onMouseOver="scrollen_stoppen()"

```



```

        onMouseOut="scrollen()"
    >
    Link
</A>
</BODY>
</HTML>

```

Beispiel für Anzeige eines Textes zu einem Hyperlink (HREF) für eine feste Zeitspanne in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var id;
    var anzeige_aktiv = false;
    var anzeige_zeit = 3000;      // 3000 Millisekunden = 3 Sekunden

    function anzeige_starten(href_text)
    {
        if(anzeige_aktiv)
        { clearTimeout(id); }

        window.status = href_text;

        id = setTimeout("anzeige_stoppen()", anzeige_zeit);

        anzeige_aktiv = true;

        return true;           // Wichtig !!!
    }

    function anzeige_stoppen ()
    {
        anzeige_aktiv = false;

        window.status = "";
    }
//-->
</SCRIPT>
</HEAD>
<BODY>
    <A HREF= ... onMouseOver="javascript:return anzeige_starten('Hinweistext...')">
    ...
    </A>
</BODY>
</HTML>

```

Beispiel zur Anzeige eines Formularfeld-Hinweises in der Statuszeile:

```

<HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    function hinweis_anzeigen(hinweis_text)
    { window.status = hinweis_text; }

    function hinweis_loeschen()
    { window.status = ""; }
// -->
</SCRIPT>
</HEAD>
<BODY>
    <FORM>
        <INPUT TYPE=TEXT
            onFocus="hinweis_anzeigen('Hinweis');"
            onBlur="hinweis_loeschen();"
        >
    </FORM>
</BODY>
</HTML>

```

Beispiel für blinkende Anzeige mit Zeitspanne von Text in der Statuszeile:

```

<HTML>

```



```

</HEAD>
<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    var zeitspanne = 6000;        // 6 Sekunden blinken lassen, danach
                                //      Statuszeile löschen
    var anzeigezeit = 500;        // 0,5 Sekunden warten nach Setzen
                                //      bzw. Löschen der Statuszeile
    var id_blinken = null;        // muss auf null initialisiert sein !!

    function blinken_timer_loeschen
    {
        if (id_blinken != null)
        {
            clearTimeout(id_blinken); // blinken stoppen falls aktiv
            id_blinken = null;
        }
    }

    function blinken_stop()
    {
        blinken_timer_loeschen;
        window.status="";
    }

    function blinken_start(status_text)
    {
        blinken_timer_loeschen;
        blinken(true, status_text); // Blinken anstossen für
                                    // paralleles Arbeiten per Timer
        setTimeout("blinken_stop()",zeitspanne); // warten und danach blinken stoppen
    }

    function blinken(ein_aus, text)
    {
        if (ein_aus)
        {
            window.status = text;
            ein_aus=false; // im nächsten Aufruf die Statuszeile löschen
        }
        else
        {
            window.status = "";
            ein_aus=true; // im nächsten Aufruf die Stautuszeile setzen
        }

        id_blinken = setTimeout("blinken(" + ein_aus + ")", anzeigezeit)
                        // nächsten Aufruf nach Ablauf der anzeigezeit starten
    }
}
-->
</SCRIPT>
</HEAD>
<BODY ... onLoad="blinken_start('Ich blinke !')">
</BODY>
</HTML>

```

STYLE direkt im HTML-Element kodierter Style (Inline-Style)
Hinweis: für Scripting ist das Style-Objekt zu nutzen

.SubFolders Zeiger auf die interne Collection FileSystemObject.Folder.Folders
Collection kann nur z.T. über das JScript-Objekt Enumerator angesprochen werden

.subtype visuelle Erscheinungsform des Überganges per .style.time2.transitionFilter Behavior-Objekt,
also visuelle Art und Weise des Überganges
optional kodierbar zur Eigenschaft .type des Behavior (bis auf 1 Ausnahme --> siehe unten)
siehe Objekt currTimeState und Behavior .style.time2
Übersicht zu Wert von .subtype bei gegebenen Wert von .type

Wert Eigenschaft .type Wert Eigenschaft .subtype

"starWipe"	"fivePoint"	muss immer kodiert sein wenn .type kodiert wurde
"barWipe"	oder "leftToRight"	
	oder "topToBottom"	
"barnDoorWipe"	oder "vertical"	
	oder "horizontal"	



"irisWipe"	oder	"rectangle" "diamond"
"ellipseWipe"		"circle"
"clockWipe"		"clockwiseTwelve"
"fanWipe"		"centerTop"
"snakeWipe"		"topLeftHorizontal"
"spiralWipe"		"topLeftClockwise"
"pushWipe"		"fromLeft"
"slideWipe"		"fromLeft"
"fade"		"crossfade"

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter1"
TYPE="barWipe"
SUBTYPE="leftToRight"
DUR="3"
TARGETELEMENT="ID_Div1"
>
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter2"
TYPE="starWipe"
SUBTYPE="fivePoint"
DUR="3"
TARGETELEMENT="ID_Div1"
>
</t:TRANSITIONFILTER>

<t:TRANSITIONFILTER ID="ID_Transfilter3"
TYPE="barnDoorWipe"
DUR="3"
TARGETELEMENT="ID_Div2"
FROM="0"
TO="1"
CALCMODE="linear"
MODE="in"
>
</t:TRANSITIONFILTER>

<DIV STYLE="height:170px;">
<DIV ID="ID_Div1"
CLASS="time_line_klasse"
STYLE="position:absolute; top:150px; left:20px; background-color:#3366CC;
padding:10px; height:80; color:white;
"
>
Test1
</DIV>

<DIV ID="ID_Div2"
CLASS="time_line_klasse"
STYLE="position:absolute; top:185px; left:60px; background-color:#FFCC00;
padding:10px; height:80;
"
>
Test2
</DIV>
</DIV>
</BODY>
</HTML>
```

.summary

Kommentar in einer Tabelle, der nicht gerendert wird
siehe Objekt table

Beispiel:



```

<TABLE ID="ID_Tabelle" BORDER CELSPACING=10 SUMMARY="Kommentar">
  <TR>
    <TD>Zelle 1</TD>
    <TD>Zelle 2</TD>
  </TR>
</TABLE>

```

.syncBehavior Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes, sinnvoll vor allem bei Time-Container (Gruppen von Objekten)

Es muss die Eigenschaft **.syncmaster** kodiert sein. Ob das Element diese Eigenschaft unterstützt, ist der Beschreibung zum jeweiligen Objekt zu entnehmen.

siehe Objekt **currentTimeState** und Behavior **.style.time2**

siehe **.syncTolerance** und **.syncmaster**

"canSlip" keine Synchronisation

sinnvoll bei Problemen der Geschwindigkeit des Internets

"locked" Element muss sich synchronisieren

sinnvoll wenn Kindelement erst animiert werden soll, wenn Elternelement komplett dazu in der Lage ist: Es wird automatisch pausiert, bis Synchronisation erreicht ist.

Beispiel für Wiedergabe per Windows Media Player:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
  <t:PAR ID="ID_Par"
    SYNCBEHAVIOR="locked"
  >
    <t:MEDIA ID="ID_Media"
      SRC="test.mid"
      PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}"
      SYNCMASTER="true"
      SYNCBEHAVIOR="locked"
      BEGIN="1"
    >
    </t:MEDIA>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
      BEGIN="ID_Media.begin"
      DUR="9"
      TARGETELEMENT="ID_Div1"
      PATH="M 0 0 L 100 300"
      FILL="freeze"
    >
    </t:ANIMATEMOTION>
    <t:SEQ ID="ID_Seq"
      STYLE="font-size:18pt;color:#ff0000"
      SYNCBEHAVIOR="locked"
      BEGIN="ID_Media.begin"
      FILL="freeze"
    >
      <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        Diese MIDI-File
      </DIV>
      <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        DUR="1.5"
      >
        wird mit dem
      </DIV>
      <DIV ID="ID_Div3"
        CLASS="time_line_klasse"
        DUR="2"
      >
        Windows Media-Player
      </DIV>
      <DIV ID="ID_Div4"

```



```

        CLASS="time_line_klasse"
        DUR="2"
    >
        abgespielt.
    </DIV>
</t:SEQ>
</t:PAR>
<DIV ID="ID_Div5"
    CLASS="time_line_klasse"
    STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
        top:70;left:10;font-size:18;border-style:solid
    ">
    >
        animierter DIV während der Wiedergabe der MIDI-File
    </DIV>
</BODY>
</HTML>

```

.syncMaster Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master **nur** genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit).
 nur sinnvoll bei Zwangssynchronisierung (siehe .syncBehavior)
 ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf !
 siehe Objekt currTimeState und Behavior .style.time2
 siehe .syncTolerance und .syncBehavior

false	Default
	Container synchronisiert nicht
true	Container muss synchronisieren

Beispiel für Wiedergabe per Windows Media Player:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY ID="ID_Body">
    <t:PAR ID="ID_Par"
        SYNCBEHAVIOR="locked"
    >
        <t:MEDIA ID="ID_Media"
            SRC="test.mid"
            PLAYER="{6BF52A52-394A-11d3-B153-00C04F79FAA6}"
            SYNCMASTER="true"
            SYNCBEHAVIOR="locked"
            BEGIN="1"
        >
        </t:MEDIA>
        <t:ANIMATEMOTION ID="ID_Animatemotion"
            BEGIN="ID_Media.begin"
            DUR="9"
            TARGETELEMENT="ID_Div1"
            PATH="M 0 0 L 100 300"
            FILL="freeze"
        >
        </t:ANIMATEMOTION>
        <t:SEQ ID="ID_Seq"
            STYLE="font-size:18pt;color:#ff0000"
            SYNCBEHAVIOR="locked"
            BEGIN="ID_Media.begin"
            FILL="freeze"
        >
            <DIV ID="ID_Div1"
                CLASS="time_line_klasse"
                DUR="1.5"
            >
                Diese MIDI-File
            </DIV>
            <DIV ID="ID_Div2"
                CLASS="time_line_klasse"
                DUR="1.5"
            >

```




```

        wird mit dem
    </DIV>
    <DIV      ID="ID_Div3"
              CLASS="time_line_klasse"
              DUR="2"
    >
        Windows Media-Player
    </DIV>
    <DIV      ID="ID_Div4"
              CLASS="time_line_klasse"
              DUR="2"
    >
        abgespielt.
    </DIV>
</t:SEQ>
</t:PAR>
<DIV      ID="ID_Div5"
          CLASS="time_line_klasse"
          STYLE="background-color:#FFCC00;position:absolute;height:40;width:120;
                top:70;left:10;font-size:18;border-style:solid
          "
    >
        animierter DIV waehrend der Wiedergabe der MIDI-File
    </DIV>
</BODY>
</HTML>

```

.syncTolerance zeitliche Toleranz in Sekunden für Zwangs-Synchronisation von Elementen auf der Timeline
 also wenn Eigenschaft **.syncBehavior** auf "locked" gesetzt ist
 sinnvoll vorallem bei Time-Container (Gruppen von Objekten)
 siehe Objekt **currTimeState** und **Behavior.style.time2**
 siehe **.syncBehavior** und **.syncMaster**
 Wert im Time-Format des Behavior
 z.B. "h:min:s.f"
 nicht id.event
 nicht id.event+zeit_wert_als_string

Beispiele:

"25:45:10"	25 Stunden, 45 Minuten, 10 Sekunden
"45:35"	45 Minuten, 35 Sekunden
"45:00.275"	45 Minuten, 0,275 Sekunde
"10.5"	10,5 Sekunden

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Anzeige()
    { alert(ID_Media.syncTolerance; }
</SCRIPT>
</HEAD>
<BODY>
    <t:PAR      ID="ID_Par"
                TIMEACTION="display"
    >
        <t:MEDIA      ID="ID_Media"
                      SCR="test.wmv"
                      BEGIN="0"
                      DUR="20s"
                      TIMEACTION="display"
                      SYNCMASTER="true"
                      SYNCBEHAVIOR="locked"
                      onmediacomplete="Anzeige()"
        >
    </t:MEDIA>
    </t:PAR>
</BODY>

```

.systemBitrate wird hier nicht erklärt

.systemCaptions wird hier nicht erklärt



`.systemLanguage` Sprache festlegen für das Objekt

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SWITCH ID="ID_Switch">
<SPAN systemLanguage="es">Spanisch</SPAN>
<SPAN systemLanguage="pt">Portugiesisch</SPAN>
<SPAN systemLanguage="en">Englisch</SPAN>
</t:SWITCH>
<t:AUDIO ID="ID_Audio"
SRC="fun.wav"
SYSTEMLANGUAGE="mi, en"
>
</ t:AUDIO>
</BODY>
</HTML>
```

`.systemLanguage` Standard-Sprache des Betriebssystems (Sprache der Installation) per navigator Objekt
navigator.systemLanguage
z.B. "en" oder "de"

`.systemLanguage` Standardsprache des Betriebssystems
Behavior .style.clientCaps
z.B. "en" oder "de"

`.systemLanguage` Standardsprache des Betriebssystems
siehe Objekt window.clientInformation
z.B. "en" oder "de"

`.systemOverdubOrSubtitle` wird hier nicht erklärt

`.tabIndex` Index des Elementes in der Tab-Tasten-Folge
für Anspringen des Dokumentes
Anspringen verbunden mit Focus erhalten
--> Ereignisse werden ausgelöst !!
unter IE 5.x
onblur, onfocus
ab IE 5.x
onblur, onfocus, onkeydown, onkeypress, onkeyup

Anspringen default per TAB-Taste
für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT,
SELECT, TEXTAREA

Anspringen default nicht per TAB-Taste
für APPLET, DIV, FRAMESET, SPAN, TABLE, TD

Beispiele:

```
<INPUT TYPE="text" TABINDEX="1">
<INPUT TYPE="text" > Deafult ist 0
<INPUT TYPE="text" TABINDEX="2">
<INPUT TYPE="submit" TABINDEX="-1">
<UL>
<LI TABINDEX="1">Tab Item 1</LI>
<LI TABINDEX="2">Tab Item 2</LI>
<LI TABINDEX="3">Tab Item 3</LI>
<LI TABINDEX="4">Tab Item 4</LI>
<LI TABINDEX="5">Tab Item 5</LI>
</UL>
```

`.tagName` Tag-Bezeichner des Objektes

Beispiel:

```
<SCRIPT>
var ID_Wert = window.prompt("Bitte ID eines Tags eingeben:");
if (ID_Wert != null)
{alert(document.all[ID_Wert].tagName) }
</SCRIPT>
```



.tagUrn Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut

Beispiel:

```
<HTML XMLNS:InetSDK='http://www.test.de'>
<STYLE>
    @media all {InetSDK\;HalloDu { behavior:url (simple.htc) }}
</STYLE>
<SCRIPT>
    function window.onload()          // überschreibt Standard window.onload-Routien !!!!
    {
        // Statuszeile als Ausgabebereich nutzen
        window.status = 'scopeName = ' + Hallo.scopeName + ' ';
        + ' tagUrn = ' + Hallo.tagUrn;
    }
</SCRIPT>
<BODY>
    <InetSDK:HelloWorld ID='Hallo'></InetSDK:HalloDu>
</BODY>
</HTML>
```

.target

Name des Ziel-Fenster bzw. Ziel-Frame

Name des Window oder Frame

bei Window laut Methode .open()

vordefinierte Namen sind:

"_blank"	neues leeres Fenster
	Fenster hat keinen Namen
"_media"	Media Bar
	ab IE 6.x
"_parent"	Elternfenster bzw. Elternframe
"_search"	Suchfenster
	ab IE 5.x
"_self"	Default
	aktuelles Fenster bzw. Frame
"_top"	oberste Fenster bzw. Frame

Wenn Zielfenster bzw. Zielframe nicht existent, soneues Fenster eröffnet mit dem Namen laut Eigenschaft .target

Beispiel:

```
<A HREF="test.htm" TARGET="_top">im obersten Fenster anzeigen</A>
```

.targetElement

ID des zu animierenden Elementes auf der Timeline

muss für Kindelement **immer** kodiert werden, wenn nicht das Elternelement animiert werden soll
muss nicht kodiert werden, wenn kein Elternelement vorliegt

Achtung: Objekt body ist das oberste Elternelement im Dokument
und somit haben Elemente innerhalb BODY immer Eltern

Empfehlung: immer kodieren

ist der Bezug des Time-Container auf das zu animierende Element

siehe Objekt currTimeState und Behavior .style.time2

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft .width) nach rechts:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function SpanInhaltInit(Kette)
    {
        // Zustand
        ID_Span1.innerText = "";

        // Kumulationsart
        ID_Span2.innerText =Kette;

        // Zaehler auf 1
        ID_Span3.innerText = "1";

        // DIV-Text festlegen
        ID_Div.innerText="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerText += " genau 1x kumulativ um ";
            ID_Div.innerText += ID_Animate.by
```



```

        ID_Div.innerText += " mal "
        ID_Div.innerText += ID_Animate.repeatCount;
    }
    else
    {
        // keine Wertkumulation von .width
        ID_Div.innerText += ID_Animate.repeatCount;
        ID_Div.innerText += " mal um ";
        ID_Div.innerText += ID_Animate.by
    }

    ID_Div.innerText += " aus";
}

function KumulationAus()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("none");
    ID_Animate.accumulate="none";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function KumulationEin()
{
    // Element darf nicht aktiv sein !!

    // ERST .accumulate definieren
    SpanInhaltInit("sum");
    ID_Animate.accumulate="sum";

    // DANACH Animation starten
    ID_Animate.beginElement();

}

function Anzeige()
{
    ID_Span1.innerText = "Animation ist beendet ";
}
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE
        ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    </t:ANIMATE>

    animierter DIV
    <DIV
        ID="ID_Div"
        CLASS="time_line_klasse"
        STYLE= "position:absolute;
            top:125px;
            left:25px;
            height:100px;
            width:125px;
            border:solid black 2px;
            "
    >
    </DIV>
    <BR>

```

Zustand der Animation:



```
<SPAN ID="ID_Span1"></SPAN>
<BR>
```

Kumulationsart:

```
<SPAN ID="ID_Span2"></SPAN>
<BR>
```

Durchlaufzaehler:

```
<SPAN ID="ID_Span3"></SPAN>
<BR>
```

```
<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
```

```
</BODY>
</HTML>
```

.text Textfarbe (Vordergrundfarbe) des Dokumentes
"rrggbb"
vordefinierte Farbname (browserspezifisch)

Beispiel:

```
<BODY ID="ID_Body" >
<BUTTON onmouseover=" ID_Body.text='green'">GREEN</BUTTON>
```

.text Text eines Objektes
nur Plaintext

.text Plain-Text im Textbereich
per textrange Objekt
Dokument muss komplett geladen sein
nur unter Windows 32-Bit

.text interner Kettenwert einer Option, also Objektes document.select.option des Objektes document.select
ändert nicht den angezeigten Text hinter <OPTION ...>
ist nur ein interner Wert und beim Formular der gesendete Wert der Option
Plain-Text

Beispiel:

```
<SCRIPT>
function Erweitern()
{
    for ( var i=0; i < ID_Select.options.length; i++)
    { ID_Select.options[i].text+=" ist eine Stadt"; }
}

function Anzeigen()
{
    var Kette="";

    for (var i=0; i < ID_Select.options.length;i++)
    {
        Kette += i
                + "angezeigter Wert ="      + ID_Select.options[i].value
                + " zu sendender Wert ="    + ID_Select.options[i].text
                + "\n";
    }

    alert(Kette);
}
</SCRIPT>
<SELECT ID="ID_Select" onchange="Anzeigen()">
  <OPTION VALUE="DE">Berlin
  <OPTION VALUE="F">Paris
  <OPTION VALUE="GB">London
</SELECT>
<INPUT TYPE=button VALUE="Alles anzeigen" onclick="Erweitern()">
```

.tfoot Zeiger auf das Objekt table.tFoot

.thead Zeiger auf das Objekt table.tHead

.timeAction Aktion des Objektes in der Timeline
Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist
siehe Objekt currTimeState und Behavior .style.time2
ab IE 5.5 "class:classname1 [classname2 ...]"
Liste mit Blanktrennung



	nur wenn Objekt aktiv, so Klassen verwendet
"display"	nur wenn Objekt aktiv, so sichtbar inaktives Objekt wird aus dem Layout entfernt Umfluss verändert sich somit Default nur für t:SEQ
"style"	nur wenn Objekt aktiv, so in Inline-Style verwendet (falls kodiert) Inline-Style per STYLE-Attribut
"visibility"	nur wenn Objekt aktiv, so sichtbar inaktives Objekt bleibt im Layout Umfluss wird nicht verändert Default außer für t:SEQ
"none"	entspricht nicht kodierte Eigenschaft

Beispiel 1:

```
<SPAN CLASS="redText boldFont"
STYLE="behavior:url(#default#time2)"
BEGIN="5"
TIMEACTION="class:redText"
>
    Dieser Text ist rot, solange Element aktiv ist
</SPAN>
```

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <P ID="ID_P1"
        CLASS="time-line_klasse"
        BEGIN="2"
        DUR="5"
    >
        Test
    </P>
    <P ID="ID_P2"
        CLASS="time_line_klasse"
        BEGIN="3"
        DUR="6"
        TIMEACTION="display"
    >
        Test
    </P>
    <P ID="ID_P3"
        CLASS="time_line_klasse"
        BEGIN="4" DUR="7"
        TIMEACTION="visibility"
    >
        Test
    </P>
    <P ID="ID_P4"
        CLASS="time_line_klasse"
        BEGIN="1"
        DUR="3"
        TIMEACTION="hidden"
    >
        Test
    </P>
    <P ID="ID_P5"
        CLASS="time_line_klasse"
        BEGIN="1"
        DUR="3"
        TIMEACTION="none"
    >
        Test
    </P>
    <H1 ID="ID_H1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="11"
        TIMEACTION="style"
        STYLE="Color:Red;"
    >
```



```

>
    Test
</H1>
</BODY>
</HTML>

```

.timeAll Zeiger auf die Collection document.body.timeAll
Feld aller Zeiger per Behavior .style.time2 verwalteter Elemente der Webseite (getimte Elemente)
siehe Objekt currTimeState und Behavior .style.time2

.timeContainer Typ der Timeline des Objektes
siehe Objekt currTimeState und Behavior .style.time2
"excl" genau ein Objekt kann agieren
"none" Default
entspricht nicht kodierter Eigenschaft
"par" alle Objekte agieren parallel
"seq" alle Objekte agieren in 1 sequentieller Folge

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <MARQUEE          ID="ID_Marquee"
                      CLASS="time_line_klasse"
                      TIMECONTAINER="seq"
                      REPEATCOUNT="indefinite"

    >
        <IMG ID="ID_Img1" CLASS="time_line_klasse" DUR="4" SRC="test1.gif" ALT="Test1">
        <IMG ID="ID_Img2" CLASS="time_line_klasse" DUR="4" SRC="test2.gif" ALT="Test2">
        <IMG ID="ID_Img3" CLASS="time_line_klasse" DUR="4" SRC="test3.gif" ALT="Test3">
    </MARQUEE>
</BODY>
</HTML>

```

.timeParent Zeiger auf das Eltern-Timeline
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    window.onload = Rekursion;    // ohne () kodieren !!

    var ZeigerAufElternTimeline = ID_Animate.timeParent;

    function Rekursion()
    { window.setInterval(Aktualisieren,100); }

    function Aktualisieren()
    {
        ID_Div.innerText = ZeigerAufElternTimeline.dur;
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV          ID="ID_Div"
                  CLASS="time_line_klasse"
                  STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"

    >
    </DIV>
    <t:EXCL ID="ID_Excl"
           CLASS="time_line_klasse"
           BEGIN="0"
           DUR="indefinite"

    >
        <t:ANIMATEMOTION          ID="ID_Animate"
                                TARGETELEMENT="ID_Div"

```




```

TO="375,0"
BEGIN="0"
DUR="3"
AUTOREVERSE="true"
REPEATCOUNT="indefinite"
>
</t:ANIMATEMOTION>
</t:EXCL>
<BUTTON ID="ID_Button" onclick="ZeigerAufElternTimeline.DUR='30s';">
    Dauer der Eltern-Timeline
</BUTTON>
<BODY>
</HTML>

```

.timeStartRule deprecated
 Startpunkt der Timeline
 "onDocLoad" Default
 Timeline beginnt nach dem kompletten Laden des Dokumentes

.title
 Fenstertitel des Dokumentes
 siehe Objekt document

Beispiel für permanenten Fenstertitel-Wechsel:

```

<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    var titel_texte_feld = [
        "Titel 1",
        "Titel 2",
        "Titel 3",
        "Titel 4"
    ];

    var wechsel_tempo = 2500;    // Zeit in ms zwischen zwei Schritten
    var zahler = 0;
    var id;

    function start()
    {
        titel_texte_feld[titel_texte_feld.length] = document.title;
        // beim ersten Aufruf wird der Original-Titel mit gespeichert
        // Länge ab 1, aber Index ab 0
        // Länge == Index +1, an desssen Position der Original-Titel abgelegt wird

        window.setTimeout("wechseln()", wechsel_tempo);
    }

    function wechseln()
    {
        document.title = titel_texte_feld [zahler];
        zahler ++;

        if(zahler >= texte.length)
        { zahler = 0 }

        id = window.setTimeout("wechseln()", wechsel_tempo);
    }
}
-->
</SCRIPT>

```

Beispiel für Fenstertitelzeile mit scrollendem Text:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
//
//      In der aktuellen Fenster-Titelzeile einen freien Text durch endloses Scrollen von links nach rechts vorsetzen
//
//*****
//
//      nachfolgende Variablen sind vom Programmierer zu setzen
//
//*****
var VorsetzText_Wert      = "Freien Text vorsetzen durch scrollen und warten ";

```



```

var ScrollGeschwindigkeit = "150"; // in Millisekunden, immer > 0, wird nicht geprüft
var DummyLeerzeichenAnzahl = 10; // immer > 0, wird nicht geprüft
// Die Leerzeichen werden nicht sichtbar angezeigt, aber gescrollt.
// So entsteht eine Wartezeit (pro Leerzeichen laut
// ScrollGeschwindigkeit)
// für die vollständige Anzeige von VorsetzText NACH dem
// vollständigem Vorsetzen per Scrollen von links nach
// rechts, ehe der Scroller wieder von vorn beginnt

/******
//
// nachfolgender Code darf nicht verändert werden
// wird anstelle von onLoad innerhalb BODY verwendet
//
/******
// rechtsbündige Auffüllung mit DummyLeerzeichen
for (i=0; i <=DummyLeerzeichenAnzahl; i++)
{VorsetzText_Wert+=" ";}

// Länge NACH Auffüllung ermitteln
var VorsetzText_Laenge=VorsetzText_Wert.length;

// globale Variablen für EndlosScrollen(), die dort laufend manipuliert werden,
// also hier initialisiert werden müssen
var Zahler="0";
var VorsetzText="";
//-->
</SCRIPT>
</HEAD>

<BODY>
<SCRIPT LANGUAGE="JavaScript1.2">

/******
//
// nachfolgender Code darf nicht verändert werden
//
/******

function EndlosScrollen()
{
    // endloses Scrollen

    if (
        (document.all)
        || (document.getElementById)
    )
    {
        // VorsetzText um nächsten Buchstaben aus dem VorsetzText_Wert erweitern
        // (einschliesslich DummyLeerzeichen)
        VorsetzText+=VorsetzText_Wert.charAt(Zahler); // charAt ab Null

        // Fenster-Titelzeile neu belegen
        document.title=VorsetzText;

        Zahler++;
        if (Zahler==VorsetzText_Laenge)
        {
            Zahler="0";
            VorsetzText="";
        }

        setTimeout("EndlosScrollen()",ScrollGeschwindigkeit);
    }
}

window.onload=EndlosScrollen; // ohne () kodieren, damit nicht sofort sondern beim Laden ausgeführt wird
//-->
</SCRIPT>
</BODY>
</HTML>

```

.title Tooltip-Text bei Mouse over über Objekt
 Plain-Text, max. 1024 Zeichen

Beispiel:



```

<SCRIPT>
    function TestSetzen(ObjektZeiger)
    {
        ObjektZeiger.title="Tooltip";
        return;
    }
</SCRIPT>
<SPAN onmouseover="TextSetzen(this)">Testtext</SPAN>

```

.title Titel der Media-Datei auf der Timeline
bei Advanced Stream Redirector (ASX)-Datei: Es wird der Text von TITLE des aktiven
Eintrages geliefert und nicht den der Datei.
Track entspricht playItem Objekt laut object.playList.item() bzw. object.playList.aktiveTrack
Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei

Beispiel 1:

```

var Kette = object.playList.item(index).title;

                index        Integer, ab 0

var Kette = object.playList.aktiveTrack.title;

```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO                    ID="ID_Video"
                               SRC="test.wmv"
                               STYLE="position:absolute;top:50px;height:100px"
    >
    </t:VIDEO>
    <SPAN    ID="ID_Span"
            STYLE="position:absolute;top:165px;"
    >
    </SPAN>
    <BUTTON                    ID="ID_Button"
                               onclick="ID_Span.innerText= ID_Video.title "
    >
        Klick
    </BUTTON>
</BODY>
</HTML>

```

Beispiel 3:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

```



```

    }
}

function AnzeigeUpdate()
{
    ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
    ID_Span3.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
    ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
    ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
    ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
    ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
    ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
    ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
}

function AnzeigeLoeschen()
{
    ID_Span1.innerText = "Titel: ";
    ID_Span2.innerText = "Autor: ";
    ID_Span3.innerText = "Abstract: ";
    ID_Span4.innerText = "Copyright: ";
    ID_Span5.innerText = "Filename: ";
    ID_Span6.innerText = "Banner: ";
    ID_Span7.innerText = "Banner Abstract: ";
    ID_Span8.innerText = "Banner MoreInfo: ";
}
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"
        TIMEACTION="visibility"
        onend="ButtonUpdate();"
        ontrackchange="AnzeigeUpdate();"
        onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
</t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON ID="ID_Button2"
        onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON ID="ID_Button3"
        onclick="ID_Media.playlist.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"

```



```

>
    Stop
</BUTTON>
</BODY>
</HTML>

```

.title Titel des StyleSheets
siehe Objekt `styleSheet`

.to Endwert zur Werterhöhung bei Elemente-Animation(en) auf der Timeline
per `.additive` oder `.accumulate`
für die Objekte `animate`, `animateMotion` und `animateColor` gilt:
`.to` wird von `.path` und `.values` überschrieben
`.to` überschreibt `.by`
Achtung: Im Objekt `animatecolor` (t:ANIMATECOLOR) ist `.values` auch als Farbliste definiert !
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel: pixelweise Ausdehnung eines DIV in seiner Breite (Style-Eigenschaft `.width`) nach rechts:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function SpanInhaltInit(Kette)
    {
        // Zustand
        ID_Span1.innerHTML = "";

        // Kumulationsart
        ID_Span2.innerHTML =Kette;

        // Zaehler auf 1
        ID_Span3.innerHTML ="1";

        // DIV-Text festlegen
        ID_Div.innerHTML="Dieser DIV dehnt sich in der Breite ";
        if (Kette == "sum")
        {
            // Wertkumulation von .width
            ID_Div.innerHTML += " genau 1x kumulativ um ";
            ID_Div.innerHTML += ID_Animate.by
            ID_Div.innerHTML += " mal "
            ID_Div.innerHTML += ID_Animate.repeatCount;
        }
        else
        {
            // keine Wertkumulation von .width
            ID_Div.innerHTML += ID_Animate.repeatCount;
            ID_Div.innerHTML += " mal um ";
            ID_Div.innerHTML += ID_Animate.by
        }

        ID_Div.innerHTML += " aus";
    }

    function KumulationAus()
    {
        // Element darf nicht aktiv sein !!

        // ERST .accumulate definieren
        SpanInhaltInit("none");
        ID_Animate.accumulate="none";

        // DANACH Animation starten
        ID_Animate.beginElement();
    }

    function KumulationEin()
    {
        // Element darf nicht aktiv sein !!

```



```

// ERST .accumulate definieren
SpanInhaltInit("sum");
ID_Animate.accumulate="sum";

// DANACH Animation starten
ID_Animate.beginElement();
}

function Anzeige()
{
    ID_Span1.innerText = "Animation ist beendet ";
}
</SCRIPT>
</HEAD>
<BODY>
    <t:ANIMATE
        ID="ID_Animate"
        TARGETELEMENT="ID_Div"
        ATTRIBUTENAME="width"
        TO="425px"
        BY="150px"
        DUR="3"
        REPEATCOUNT="3"
        BEGIN="indefinite"
        FILL="freeze"
        onend="Anzeige();"
        onrepeat="ID_Span3.innerText= ID_Animate.currTimeState.repeatCount + 1;"
    >
    <t:ANIMATE>

animierter DIV
<DIV ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE= "position:absolute;
            top:125px;
            left:25px;
            height:100px;
            width:125px;
            border:solid black 2px;
            "
>
</DIV>
<BR>

Zustand der Animation:
<SPAN ID="ID_Span1"></SPAN>
<BR>

Kumulationsart:
<SPAN ID="ID_Span2"></SPAN>
<BR>

Durchlaufzaehler:
<SPAN ID="ID_Span3" ></SPAN>
<BR>

<BUTTON ID="ID_Button1" onclick="KumulationAus()">Kumulation aus </BUTTON>
<BUTTON ID="ID_Button2" onclick="KumulationEin()">Kumulation ein</BUTTON>
</BODY>
</HTML>

```

.to

Ende-Prozentsatz des kompletten Überganges bei Ende der Animation des Übergangsfilters
per .style.time2.transitionFilter Behavior-Objekt
Ende des Übergangsfilters mit nur teilweise vollendetem Übergang
nicht zusammen mit Eigenschaften .by und .value kodieren, sonst wird .to ignoriert
siehe Objekt currTimeState und Behavior .style.time2
Ziffernfolge Floating point
Wert numerisch von 0 bis 1
Standard ist "1.0" oder "1"
0 entspricht 0 % des kompletten Überganges
1 entspricht 100% des kompletten Überganges
Bsp.: 0.3 entspricht: Mit **nur** 30 % des kompletten Überganges die Animation beenden

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
```



```

<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:TRANSITIONFILTER ID="ID_Transfilter1"
FROM="0.3"
TO="0.7"
TYPE="barWipe"
DUR="3"
TARGETELEMENT="ID_Div"
>
</t:TRANSITIONFILTER>

<DIV ID="ID_Div"
CLASS="time_line_klasse"
DUR="9"
STYLE=" width:400px; height:100px;background:#CC3333;color:#FFFFFF;"
>
<SPAN ID="ID_Span"
STYLE="position:relative; top:20px;margin:160;"
>
Test
</SPAN>
</DIV>
</BODY>
</HTML>

```

.toElement Referenz auf Maus-Eventauslösendes Ziel-Element bei Mausebewegung
nicht für Event ondragleave
Objekt event

Beispiel:

```

<SCRIPT>
function Aendern()
{ ID_Span.innerHTML=window.event.toElement.tagName; }
</SCRIPT>
<SPAN onmouseout="Aendern()">
<P>Ueberfahre mich mit Maus</P>
<P>Zielelement =
<SPAN ID="ID_Span"></SPAN>
</P>
</SPAN>

```

.top obere Pixelposition des Rechteckes um ein Objekt
auch textrectangle Objekt

Beispiel für Nicht-TextRectangle-Objekt:

```

<SCRIPT>
function Anzeigen(ObjektZeiger)
{
var Rechteck = ObjektZeiger.getBoudingClientRect();

alert( "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
+ "\n"
+ "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
);
}
</SCRIPT>
<P onclick="Anzeigen(this)">

```

Beispiel für TextRectangleObjekt:

```

<SCRIPT>
function Anzeigen(ObjektZeiger)
{
var TextRectangleCollection = document.body.ObjektZeiger.TextRange.getClientRects();

// .getClientRects() liefert immer Colletion der textrectangle Objekte !!!

var Rechteck = TextRectangleCollection[0];

```




```

        alert(      "oben links (x,y) = " + Rechteck.left + " " + Rechteck.top
                    + "\n"
                    + "unten rechts (x,y) = " + Rechteck.right + " " + Rechteck.bottom
                    );
    }
</SCRIPT>
<BODY>
<DIV onclick="Anzeigen(this)">
    Test
</DIV>
</BODY>

```

.top	Zeiger auf das oberste Fenster in der Fenster-Hierarchie siehe Objekt window
.topMargin	Top Margin in Pixel des Dokumentes Margin: Abstand des Aussenrandes eines Objektes zur Umgebung Padding: Abstand des Objekthinhaltes zum Aussenrand document.body.topMargin
.TotalSize	Gesamten Speicher in Bytes auf Laufwerk ermitteln
Beispiel:	<pre> var DateiSystem = new ActiveXObject("Scripting.FileSystemObject"); var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:"); var Laufwerk = DateiSystem.GetDrive(Laufwerk_Bezeichnung); var Laufwerk_VolumeName = Laufwerk.VolumeName; var Laufwerk_TotalerPlatz = Laufwerk.TotalSize; alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalerPlatz); </pre>
TRANSITIONTYPE	wird hier nicht erklärt, da Komponenten von MS DirectMusic-Software
.trueSpeed	zeitliche Takteinheit zur Erzeugung der Pause laut Eigenschaft .scrollDelay eines marquee Objektes false Default Takteinheit = 60 Millisekunden = 1 Sekunde wenn .scrollDelay < 60 so Sekundentakt wenn .scrollDelay > 59 so Millisekundentakt true Takteinheit = immer 1 Millisekunde
.type	Bezeichner des Events, also Eventart Objekt event ist immer der Eventname OHNE den Präfix "on" Bsp. Kette ist "click" für das Ereignis onclick
.type	MIME-Typ des Objektes (Multipurpose Internet Mail Extension) wenn die Eigenschaft .classid nicht kodiert wird, dann immer .type kodieren bzw. die im Browserstandard enthaltenen MIME verwenden Hinweis: MIME wird von Simple Mail Transfer Protocol (SMTP) benutzt z.B. bei Audio, Images, Video, Texten
.type	Art des Buttons (Standard-Behavior (Verhalten) des Buttons) "button" Default. Kommando-Button "reset" Reset-Button wenn im Formular so Funktionalität für Formular-Reset analog zum input-reset-Objekt Achtung: zugleich NAME-Attribut kodieren ! "submit" Submit-Button wenn im Formular so Funktionalität für Formular-Send analog zum input-submit-Objekt (senden der Werte laut .name und .value) Achtung: zugleich NAME-Attribut kodieren !
.type	Typ (Variante) des Input-Objektes (Art der Control-Elemente vom gemeinsamen Typ INPUT) z.B. im Formular Achtung: Soll der Wert eines Control-Elementes im Formular gesendet werden, so muss für das Element das NAME-Attribut kodiert sein (Wert des Name-Attributs muss elementspezifisch sein: siehe z.B. Checkbox-Control-Element). Gesendet werden in der Regel Wert und Name. Das Name-Attribut ist wichtig für die Auswertung des Wertes des Control-Elementes: Name-Attribut hat dazu die Funktionalität wie das ID-Attribut. Hinweis: Wenn Control-Element ein ID-Attribut erlaubt, dann ist ID auch zur Referenzierung verwendbar z.B. wert_laut_id_attribut.value



Hinweis:	Wert des Control-Elementes
	nur Stringwerte möglich
	Standardwerte laut Eigenschaft .defaultValue
	(Eigenschaft nur per Script nutzbar, da keine HTML-Attribut verfügbar)
	INPUT type=checkbox on
	INPUT type=reset Reset
	INPUT type=submit Submit Query
	alle anderen Input-Elemente haben keinen Standardwert
Wert laut .value (auch HTML-Attribut vorhanden)	
	folgenden Werte sind sendbar:
	INPUT type=checkbox selektierter Wert
	INPUT type=file Dateiname laut Eingabe
	INPUT type=hidden selektierter Wert einer Box-Control (selektierte Option)
	INPUT type=password Eingabewert
	INPUT type=radio selektierter Wert
	INPUT type=reset das Label des Elementes (falls Label existent ist)
	INPUT type=submit das Label des Elementes (falls Label existent ist)
	INPUT type=text Eingabewerte
	Werte sind les- und schreibbar
	nur lesen bei INPUT type=file
kann in " " kodiert werden, muss aber nicht	
"button"	Button-Control
"checkbox"	Checkbox-Control
	jedes Element mit seinem spezifischen Namen
	erhält intern eine fortlaufende Nummer
	Achtung: Wenn Elemente des Checkbox-Control
	mit gemeinsamen Namen, so
	haben diese eine gemeinsame Nummer.
	Mehrfachauswahl (Mehrfachselektion) möglich
"file"	Objekt zum Upload einer Datei
	(Upload aus Sicht des Client = Browser
	Download aus Sicht des PC-User und seiner Festplatte)
	Die Datei kann auf dem PC gespeichert werden
	(Browser öffnet automatisch eine
	Dialogbox zur Angabe der Speicherortes).
"hidden"	Hidden-Control
"image"	Image-Control das angeklickt werden kann
	(Alternative zu einem Link mit Bild)
	Achtung: Es löst sofort Senden des Formulars aus,
	falls Ereignis onclick nicht abgefangen wird !
	gesendet wird name.x
	name.y
	mit name als Wert des Attributes NAME
	x und y als Pixelkoordinaten der
	linken oberen Ecke des Bildes
	gesendet wird nicht der Wert laut VALUE-Attribut !
"password"	Text-Control mit veränderter Anzeige eingegebener
	Zeichen: Anzeige durch Dummyzeichen
"radio"	Radio-Button-Control
	Gruppierung möglich: Gruppe = Radio-Button-
	Controls mit gemeinsamen Namen
	jede Gruppe hat einen spezifischen Namen
	gesendet wird nur der Wert des selektierten
	Radio-Button-Control laut
	VALUE-Attribut
"reset"	Button-Control mit Funktionalität der Initialisierung
	des Formulars auf Standardwerte
	Label möglich oder per VALUE-Attribut kodierbar
	Eventbehandlung: siehe Formular
"submit"	Button-Control mit Funktionalität des Sendens des Formulars
	Label möglich oder per VALUE-Attribut kodierbar
	Eventbehandlung: siehe Formular
"text"	Default
	einzeiliges Texteingabe-Control
.type	Type des TEXTAREA-Control
	siehe textarea Objekt



.type	Mimetype des Scriptes per script Objekt Mimetype wird von der Scriptmaschine zum Parsen verwendet Achtung: Wert muss passend zum Wert der Eigenschaft .language sein ! "text/ecmascript" ECMA-Script "text/JScript" Microsoft JScript "text/javascript" JavaScript "text/vbs" VB-Script "text/vbscript" VB-Script (wie text/vbs) "text/xml" XML.
.type	prüfen auf multiples select Objekt "select-multiple" multiple Auswahl möglich Attribut .multiple ist true "select-one" Default keine multiple Auswahl Attribut .multiple ist false (Standard für .multiple)
.type	Type der Selektion per document.selection Objekt des Dokumentes siehe auch Methoden .createRange() .createControlRange() und .createTextRange() Objekt textrange "none" keine Selektion vom Typ text oder control wird auch geliefert, wenn Cursor auf einen leeren Bereich der Webseite gesetzt wird "text" Text wurde selektiert (siehe document.selection.textrange Collection) typisch für Markierung von Text auf Webseite, um diesen dann in den notepad zu laden "control" Control-Element(e) wurde(n) selektiert z.B. Elemente zum Resize des Dokumentes (siehe document.selection.controlRange Collection)

Beispiel:

```
<BODY onclick="alert(document.selection.type)">
TestText
</BODY>
```

.type	Sprache des Cascading Style Sheets (CSS) des style Objektes																																																																																																												
.type	MIME-Typ eines Media-Elementes auf der Timeline Media-Datei zum Element laut Eigenschaft .src																																																																																																												
.type	visuelle Erscheinungsform des Überganges per .style.time2.transitionFilter Behavior-Objekt, also visuelle Art und Weise des Überganges optional kodierbar ist zusätzlich die Eigenschaft .subtype des Behavior (bis auf 1 Ausnahme --> siehe unten) siehe Objekt currTimeState und Behavior .style.time2 Übersicht zu Wert von .subtype bei gegebenen Wert von .type																																																																																																												
	<table><tr><th><u>Wert</u></th><th><u>Eigenschaft</u></th><th><u>.type</u></th><th><u>Wert</u></th><th><u>Eigenschaft</u></th><th><u>.subtype</u></th></tr><tr><td>"starWipe"</td><td></td><td></td><td>"fivePoint"</td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td>muss immer kodiert sein</td><td></td></tr><tr><td></td><td></td><td></td><td></td><td>wenn .type kodiert wurde</td><td></td></tr><tr><td>"barWipe"</td><td></td><td></td><td>"leftToRight"</td><td></td><td></td></tr><tr><td></td><td></td><td>oder</td><td>"topToBottom"</td><td></td><td></td></tr><tr><td>"barnDoorWipe"</td><td></td><td></td><td>"vertical"</td><td></td><td></td></tr><tr><td></td><td></td><td>oder</td><td>"horizontal"</td><td></td><td></td></tr><tr><td>"irisWipe"</td><td></td><td></td><td>"rectangle"</td><td></td><td></td></tr><tr><td></td><td></td><td>oder</td><td>"diamond"</td><td></td><td></td></tr><tr><td>"ellipseWipe"</td><td></td><td></td><td>"circle"</td><td></td><td></td></tr><tr><td>"clockWipe"</td><td></td><td></td><td>"clockwiseTwelve"</td><td></td><td></td></tr><tr><td>"fanWipe"</td><td></td><td></td><td>"centerTop"</td><td></td><td></td></tr><tr><td>"snakeWipe"</td><td></td><td></td><td>"topLeftHorizontal"</td><td></td><td></td></tr><tr><td>"spiralWipe"</td><td></td><td></td><td>"topLeftClockwise"</td><td></td><td></td></tr><tr><td>"pushWipe"</td><td></td><td></td><td>"fromLeft"</td><td></td><td></td></tr><tr><td>"slideWipe"</td><td></td><td></td><td>"fromLeft"</td><td></td><td></td></tr><tr><td>"fade"</td><td></td><td></td><td>"crossfade"</td><td></td><td></td></tr></table>	<u>Wert</u>	<u>Eigenschaft</u>	<u>.type</u>	<u>Wert</u>	<u>Eigenschaft</u>	<u>.subtype</u>	"starWipe"			"fivePoint"							muss immer kodiert sein						wenn .type kodiert wurde		"barWipe"			"leftToRight"					oder	"topToBottom"			"barnDoorWipe"			"vertical"					oder	"horizontal"			"irisWipe"			"rectangle"					oder	"diamond"			"ellipseWipe"			"circle"			"clockWipe"			"clockwiseTwelve"			"fanWipe"			"centerTop"			"snakeWipe"			"topLeftHorizontal"			"spiralWipe"			"topLeftClockwise"			"pushWipe"			"fromLeft"			"slideWipe"			"fromLeft"			"fade"			"crossfade"		
<u>Wert</u>	<u>Eigenschaft</u>	<u>.type</u>	<u>Wert</u>	<u>Eigenschaft</u>	<u>.subtype</u>																																																																																																								
"starWipe"			"fivePoint"																																																																																																										
				muss immer kodiert sein																																																																																																									
				wenn .type kodiert wurde																																																																																																									
"barWipe"			"leftToRight"																																																																																																										
		oder	"topToBottom"																																																																																																										
"barnDoorWipe"			"vertical"																																																																																																										
		oder	"horizontal"																																																																																																										
"irisWipe"			"rectangle"																																																																																																										
		oder	"diamond"																																																																																																										
"ellipseWipe"			"circle"																																																																																																										
"clockWipe"			"clockwiseTwelve"																																																																																																										
"fanWipe"			"centerTop"																																																																																																										
"snakeWipe"			"topLeftHorizontal"																																																																																																										
"spiralWipe"			"topLeftClockwise"																																																																																																										
"pushWipe"			"fromLeft"																																																																																																										
"slideWipe"			"fromLeft"																																																																																																										
"fade"			"crossfade"																																																																																																										

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
```



```

<BODY>
  <t:TRANSITIONFILTER ID="ID_Transfilter1"
    TYPE="barWipe"
    SUBTYPE="leftToRight"
    DUR="3"
    TARGETELEMENT="ID_Div1"
  >
</t:TRANSITIONFILTER>

  <t:TRANSITIONFILTER ID="ID_Transfilter2"
    TYPE="starWipe"
    SUBTYPE="fivePoint"
    DUR="3"
    TARGETELEMENT="ID_Div1"
  >
</t:TRANSITIONFILTER>

  <t:TRANSITIONFILTER ID="ID_Transfilter3"
    TYPE="barnDoorWipe"
    DUR="3"
    TARGETELEMENT="ID_Div2"
    FROM="0"
    TO="1"
    CALCMODE="linear"
    MODE="in"
  >
</t:TRANSITIONFILTER>

  <DIV STYLE="height:170px;">

    <DIV ID="ID_Div1"
      CLASS="time_line_klasse"
      STYLE="position:absolute; top:150px; left:20px; background-color:#3366CC; padding:10px;
        height:80; color:white;
      "
    >
      Test1
    </DIV>

    <DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      STYLE="position:absolute; top:185px; left:60px; background-color:#FFCC00;
        padding:10px; height:80;
      "
    >
      Test2
    </DIV>
  </DIV>
</BODY>
</HTML>

```

.type Mimetype des StyleSheets
siehe Objekt styleSheet

.Type Beschreibung zum Suffix des Ordnerbezeichners liefern laut Registry

Beispiel:

```

var OrdnerName                = "c:\\windows\\desktop\\";
var DateiSystem              = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                    = DateiSystem.GetFolder(OrdnerName);
alert(Ordner.Type);

```

.Type Beschreibung zum Suffix des Dateibezeichners liefern laut Registry
z.B. Suffix ist .TXT Beschreibung ist "Text Document"

Beispiel:

```

var DateinameMitPfad         = "c:\\test.txt";
var DateiSystem              = new ActiveXObject("Scripting.FileSystemObject");
var Datei                    = DateiSystem.GetFile(DateinameMitPfad);
alert(Datei.Type);

```

.typeDetail Type der Selektion per document.selection Objekt des Dokumentes, wobei die Hostanwendung diese Eigenschaft unterstützen und mit Wert belegen muss

.uniqueID durch den Browser automatisch-generiertes ID des Objektes
Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde



kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden

Beispiel:

```
<PUBLIC:ATTACH EVENT="onload" FOR="window" ONEVENT="init()"/>
<SCRIPT LANGUAGE="JScript">
    function init()
    {
        var newTextAreaID = element.document.uniqueID;
        element.document.body.insertAdjacentHTML ( "beforeEnd",
            "<P><TEXTAREA STYLE='height: 200 ;'"
            + "width: 350' ID=" + newTextAreaID
            + "></TEXTAREA></P>"
        );
    }
</SCRIPT>
```

UNSELECTABLE

Selektionsfähigkeit eines Objektes

"off" Default. selektierbar

"on" nicht selektierbar

Beispiel:

```
<SPAN UNSELECTABLE="on" >
    Dieser Text kann nicht selektiert werden
    <TEXTAREA WRAP="PHYSICAL" ROWS="5"
        STYLE="font-weight: bold;"
    >
        Dieser Text kann selektiert werden
    </TEXTAREA>
    Dieser Text kann nicht selektiert werden
</SPAN>
```

.updateInterval

Refresh-Intervall des Bildschirmes: aus dem Puffer neu schreiben
per screen Objekt

.updateMode

Art des Updates der Eigenschaften eines Elementes nach dem Start des Elementes auf der Timeline
folgende Eigenschaften können geupdatet werden:

.autoReverse

.begin

.dur

.end

.fill

.repeatCount

.repeatDur

.speed

siehe Objekt currTimeState und Behavior .style.time2

"Auto" automatisches Update

immer ausgeführt per Methode .beginElement()

"Reset" Default

Reset auf Initialwert nach einer Veränderung

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<SCRIPT>
    function Aendern()
    {
        ID_Video.updateMode = ID_Select1.options.value;
        ID_Video.speed      = ID_Select1.options.value;
    }
</SCRIPT>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        UPDATEMODE="reset"
        STYLE="width:175px; height:150px;"
    >
    </t:VIDEO>
    <BR>
    updateMode waehlen:
    <SELECT NAME="ID_Select1">
```



```

        <OPTION VALUE="auto">Auto</OPTION>
        <OPTION VALUE="reset" SELECTED>Reset</OPTION>
    </SELECT>
    <BR>
    Geschwindigkeit waehlen:
    <SELECT NAME="ID_Select2">
        <OPTION VALUE="0.25">25%</OPTION>
        <OPTION VALUE="0.50">50%</OPTION>
        <OPTION VALUE="0.75">75%</OPTION>
        <OPTION VALUE="1" SELECTED>100% </OPTION>
        <OPTION VALUE="2" SELECTED>200% </OPTION>
    </SELECT>
    <BR>
    <BUTTON ID="ID_Button1" onClick="Aendern();">
        Geschwindigkeit aendern
    </BUTTON>
    <BUTTON ID="ID_Button2" onClick="document.body.beginElement();">
        Restart
    </BUTTON>
</BODY>
</HTML>

```

.URL	Url des Dokumentes Achtung: Gross-und kleinschreibung wird unterschieden !!! ist identisch mit location.href
.URL	Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-Datei von Element auf der Timeline es muss Ereignis onURLFlip aufgetreten sein siehe Objekt currTimeState und Behavior .style.time2 window.event.URL
.URLUnencoded	Url des Dokumentes ohne Zeichenencoding
.urn	Uniform Resource Name (URN) des Dokumentes
.useMap	Url oder Anker für client-seitige Image-Map
.userAgent	Browser-Codename und –Version per navigator Objekt navigator.userAgent z.B. IE 6.0 liefert unter Windows XP "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"

Beispiel 1:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        if(navigator.userAgent.indexOf("Windows NT 5.1")>-1)
        { ..... }
    }
</SCRIPT>

```

Beispiel 2 für prüfen auf ActiveX beim IE:

```

var ActiveXAktiv = false;

var NavigatorObjekt = window.navigator;           // Zeiger

var BrowserArt     = NavigatorObjekt.userAgent;    // String
var IEerkannt      = (BrowserArt.indexOf('IE') > -1); // true, so IE erkannt

var Plattform      = NavigatorObjekt.platform;     // String
var Win32Erkannt   = (Plattform == "Win32");       // true, so Win32-Bit erkannt

ActiveXAktiv       = (IEerkannt && Win32Erkannt);  // true, so IE und Win32-Bit erkannt,
                                                    // also ActiveX möglich

```

.userAgent	HTTP User-Agent siehe Objekt window.clientInformation z.B. IE 6.0 liefert unter Windows XP "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
------------	---

.userLanguage	vom User eingestellte Sprache des Betriebssystems (nicht die Sprache der Installation,
---------------	--



	sondern die regionale Sprache des Betriebssystems)
	per navigator Objekt navigator.userLanguage z.B. "en" oder "de"
.userLanguage	Sprache des Betriebssystems laut Usereinstellung Behavior .style.clientCaps z.B. "en" oder "de"
.userLanguage	Sprache des Betriebssystems laut Usereinstellung siehe Objekt window.clientInformation z.B. "en" oder "de"
.userProfile	Zeiger auf Collection userProfile siehe Objekt navigator
.vAlign	Lage der CAPTION einer Tabelle es darf nur 1 CAPTION zur Tabelle existieren nach Änderung ist ein Tabellen-Refresh per Methode .refresh() notwendig siehe Objekt table.caption "top" Überschrift am Kopf der Tabelle Standard "bottom" Überschrift am Fuss der Tabelle
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf
.value	Wert eines Objekt-Attributes Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar

Beispiel für Input-Objekt und seine Varianten:

für Input-Elemente (input Objekt) gelten folgende Standardwerte:

INPUT type=checkbox	on
INPUT type=reset	Reset
INPUT type=submit	Submit Query

alle anderen Input-Elemente haben keinen Standardwert

für Input-Elemente (input Objekt) sind folgenden Werte sendbar:

INPUT type=checkbox	selektierter Wert
INPUT type=file	Dateiname laut Eingabe
INPUT type=hidden	selektierter Wert einer Box-Control (selektierte Option)
INPUT type=password	Eingabewert
INPUT type=radio	selektierter Wert
INPUT type=reset	das Label des Elementes (falls Label existent ist)
INPUT type=submit	das Label des Elementes (falls Label existent ist)
INPUT type=text	Eingabewerte

Werte sind les- und schreibbar
nur lesen bei INPUT type=file

Beispiel für option objekt eines select objektes:

```

<SCRIPT>
function Anzeigen()
{
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
}
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
    <OPTION VALUE="1">Auswahl 1 </OPTION>
    <OPTION VALUE="2">Auswahl 2 </OPTION>
    <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Beispiel für Zeichenkette auf Wertebereich der Zeichen prüfen:




```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function pruefe_zeichenkette(zeichenkette, wertebereich)
{
    // wertebereich ist Zeichenkette z.B. "0123456789 -+/,() "

    var rueckgabewert = true;        // Annahme: Zeichenkette hat NUR Zeichen
                                     //                                     aus dem Wertebereich

    var zeichen_aus_zeichenkette;

    //      Zeichenkette zeichenweise analysieren: Jedes Zeichen mit Wertebereich vergleichen
    for (var i = 0; i < zeichenkette.length; i++)
    {
        zeichen_aus_zeichenkette = zeichenkette.charAt(i);

        if (wertebereich.indexOf(zeichen_aus_zeichenkette) == -1)
        {
            rueckgabewert = false;
            break;    // ungültiges Zeichen gefunden, also abbrechen
        }
    }

    return rueckgabewert;
}

function pruefe_eingabe(zu_pruefende_zeichenkette)
{
    if (pruefe_zeichenkette(zu_pruefende_zeichenkette, "0123456789 -+/,() ") )
    {alert("Eingabe ist korrekt !");}
    else
    {alert("Eingabe ist nicht korrekt !"); }
}
//-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>

    Telefon:
    <INPUT  TYPE="text"
            NAME="Telefon"
            VALUE=""

    >
    <INPUT  TYPE="button"
            VALUE="Ueberpruefen"
            onclick="pruefe_eingabe(this.form.Telefon.value)">

</FORM>
</BODY>
</HTML>

```

.values

2D-Animation bei der Elemente-Animation(en) auf der Timeline
per .additive oder .accumulate
anhand absoluter Koordinaten im Grafiksystem
siehe .path für Vektorgraphik-Animation anhand relativer und absoluter Koordinaten
Werteliste korrespondiert zu der Werteliste der Eigenschaft .keyTimes
benötigt .calcMode auf "spline" .keySplines und .keyTimes
für die Objekte animate, animateMotion und animateColor gilt:
.values wird von .path überschrieben
.values überschreibt .by .from .to
Achtung: Im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
    .div_Klasse{position:absolute; top:195px; height:100px; width:150px; border:solid black 2px;}
</STYLE>
</HEAD>
<BODY>
    <SPAN  ID="ID_Span1"
            CLASS="time_line_klasse"
            DUR="1"

```



```

        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=( ID_Animatemotion.currTimeState.simpleTime);"
    >
        0
    </SPAN>
    <BR>
    <SPAN ID=" ID_Span2"
        CLASS="time_line_klasse"
        DUR=".1"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=( ID_Animatemotion.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <DIV ID="ID_Div"
        CLASSE="div_klasse"
    >
        animierter Div
    </DIV>
    <t:ANIMATEMOTION ID="ID_Animatemotion"
        BEGIN="1; ID_Button.click+1"
        DUR="6s"
        AUTOREVERSE="true"
        CALCMODE="spline"
        KEYSPLINES="0 1 .5 1;.5 1 0 1"
        KEYTIMES="0;.5;1"
        VALUES="25,0;250,50;500,0"
        TARGETELEMENT="ID_Div"
        FILL="hold"
    >
    </t:ANIMATEMOTION>
    <BUTTON ID="ID_Button">Restart</BUTTON>
</BODY>
</HTML>

```

.values

Farbwerte für die Elemente-Animation(en) auf der Timeline
 per .additive oder .accumulate
 nur für Objekt animatecolor (t:ANIMATECOLOR)
 Achtung: Nur im Objekt animatecolor (t:ANIMATECOLOR) ist .values auch als Farbliste definiert !
 Es existieren als 2 kodierbare Varianten von .values.
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
    <t:PAR ID="ID_Par"
        BEGIN="0"
        DUR="10"
        FILL="hold"
    >
        <t:ANIMATECOLOR ID="ID_Animatecolor1"
            TARGETELEMENT="ID_Div1"
            ATTRIBUTENAME="background-color"
            VALUES="#0000FF;cyan"
            BEGIN="0"
            DUR="5"
            FILL="hold"
        >
        </t:ANIMATECOLOR>
        <t:ANIMATECOLOR ID="ID_Animatecolor2"
            TARGETELEMENT="ID_Div2"
            ATTRIBUTENAME="background-color"
            VALUES="cyan ;#0000FF"
            BEGIN="0"
            DUR="5"
            FILL="hold"
        >
    >

```



```

</t:PAR>
<DIV ID="ID_Div1"
      CLASS="time_line_Klasse"
      STYLE="position: absolute; left: 68px; width: 279px; top: 260px; height: 217px;
            border: 1px solid black; background-color: green;
            "
      >
        animierter Div
</DIV>
<DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      STYLE="position: absolute; left: 112px; width: 188px; top: 318px; height: 98px;
            padding-left: 3; background-color: gray;
            "
      >
        animierter Div
</DIV>
</BODY>
</HTML>

```

.values

Animationsschritte als Prozentsatz des kompletten Überganges
 per .style.time2.transitionFilter Behavior-Objekt
 nicht zusammen mit Eigenschaften .by und .to und .from kodieren, da diese sonst ignoriert werden
 siehe Objekt currTimeState und Behavior .style.time2
 Liste aus per Semikolon getrennten Elementen
 Element:
 Ziffernfolge Floating point
 Wert numerisch von 0 bis 1
 0 entspricht 0 % des kompletten Überganges
 1 entspricht 100% des kompletten Überganges
 Elementefolge: muss numerisch aufsteigend sein
 erstes Element muss nicht 0.0 sein
 letztes Element muss nicht 1.0 sein

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>

  <t:TRANSITIONFILTER ID="ID_Transfilter1"
    BEGIN="ID_Div1.begin"
    TYPE="barWipe"
    DUR="5"
    TARGETELEMENT="ID_Div1"
    VALUES=".1;.17;.27;.37;.47;.56;.65;.71;.82;.92;1.0"
    CALCMODE="discrete"
  >
</t:TRANSITIONFILTER>

  <t:TRANSITIONFILTER ID="ID_Transfilter2"
    BEGIN="ID_Div1.begin"
    TYPE="barWipe"
    DUR="5"
    TARGETELEMENT="ID_Div2"
    VALUES=".1;.17;.27;.37;.47;.56;.65;.71;.82;.92;1.0"
    CALCMODE="linear"
  >
</t:TRANSITIONFILTER>
<BR>
<INPUT TYPE="button" ID="ID_Button" VALUE="Start Transition">

  <DIV ID="ID_Div1"
    CLASS="time_line_Klasse"
    BEGIN="ID_Button.click"
    DUR="indefinite"
    STYLE="position: relative; left: 20px; width: 420px; height: 100px;
          background-image: url(test.gif); background-repeat: no-repeat;
          "
  >
</DIV>

```



```

<DIV ID="ID_Div2"
CLASS="time_line_Klasse"
BEGIN="ID_Button.click"
DUR="indefinite"
STYLE="position:relative; left:20px; width:420px; height:100px;
background-image:url(test.gif); background-repeat: no-repeat;
"
>
</DIV>
</BODY>
</HTML>

```

.vcard_name Werte für vCard für Autocomplete (Autovervollständigung) bei Formular per Text-Control
 z.B. input text Objekt
 anhand vCard-Schematas analog zur Visitenkarte
 Userdaten werden als Profil gespeichert
 bei aktivem Autocomplete wird der Wert vom NAME-Attribut
 nicht verwendet wenn VCARD_NAME ist kodiert
 verwendet wenn VCARD_NAME nicht ist kodiert
 bei inaktivem Autocomplete werden
 weder der Wert des NAME-Attribut
 noch der Wert von VACARD_NAME verwendet
 siehe auch Eigenschaft .autocomplete

Beispiel: Der in die Textbox eingegebenen Wert wird als Email-Adresse aufgefasst
 (und gespeichert falls Autocomplete aktiv ist)
 es wird nicht "CustomerEmail" gesendet sondern der Textboxinhalt als vCard.Email-Schema

```

<INPUT TYPE = text
NAME= "CustomerEmail"
VCARD_NAME = "vCard.Email"
>

```

.version Document Type Definition-Version (DTD-Version)
 ab IE 5.x

.vLink Farbe eine VLINK des Dokumentes
 document.body.vLink
 "#rrggbb"
 vordefiniertes Farbname (browserspezifisch)

.vlinkColor Farbe bereits geklickter Links des Dokumentes
 document.body.vlinkColor
 "#rrggbb" Standard "#800080"
 vordefinierter Farbname (browserspezifisch)

.volume aktuelle Wiedergabe-Lautstärke (Run time volume) per Objekt bgsound, wenn die Media-Datei
 nicht selbst Lautstärke-Einstellungen während der Wiedergabe vornimmt:
 Bsp.: MIDI Volume ohne Wirkung
 WAVE Volume soll Wirkung haben
 Hinweis: eventuelle Veränderung der Regler in der Windows-Lautstärke-Regelung beachten.
 -10 bis 0
 0 maximal laut

```

<HEAD>
<SCRIPT>
function KanalVerteilung(Wert)
{ ID_bgsound.balance = Wert;}

function GenauEinmal()
{
    ID_bgsound.loop = 1;
    ID_bgsound.src = ID_bgsound.src; // restart
}

function Endlos()
{
    ID_bgsound.loop = -1;
    ID_bgsound.src = ID_bgsound.src; // restart
}
</SCRIPT>
<BGSOUND ID="ID_bgsound" SRC="sound.wav">
</HEAD>
<BODY>

```



```

<BUTTON onclick="GenauEinmal()"></BUTTON>
<BUTTON onclick="Endlos()"></BUTTON>
<BUTTON onclick="KanalVerteilung(-10)"></BUTTON>
<BUTTON onclick="KanalVerteilung(10)"></BUTTON>
<BUTTON onclick="KanalVerteilung (0)"></BUTTON>
<B>Volume control:</B>&nbsp;
<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-10;"
>Mute

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-7;"
>25% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
        onpropertychange="ID_bgsound.volume=-5;"
>50% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=-2;"
>75% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_bgsound.volume=0;"
>100% Volume
</BODY>

```

.volume

Lautstärke eines Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes für alle Audio-Elemente des body Objektes auch möglich
 Achtung: Bei Verwendung von DirectMusic wirkt sich das auf den DirectMusic-Player aus !
 siehe Objekt currTimeState und Behavior .style.time2
 siehe Objekt bgsound
 Prozent der Lautstärke der Eltern
 0 bis 100
 0 stumm
 100 Lautstärke wie die der Eltern

Beispiel für Verwendung von DirectMusic:

```
<t:AUDIO VOLUME="100" PLAYER=dmusic SRC="sample.sgt">
```

Beispiel für Abspielen einer Sounddatei per Timeline als Alternative zum Objekt bgsound:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:MEDIA ID="ID_Media"
        BEGIN="indefinite;"
        SRC="test.wmv"
        FILL="remove"
        onmediacomplete="Timer2.beginElement();"
>
</t:MEDIA>

<BR>

<BUTTON onclick="ID_Media.beginElement();">Start</BUTTON>
<BUTTON onclick="ID_Media.endElement();">Stop</BUTTON>

<BR>

<B>Volume control:</B>&nbsp;

<INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_Media.mute='true';"
>Mute

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
        onpropertychange="ID_Media.mute='false'; ID_Media.volume=25;"
>25% Volume

```



```

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe" CHECKED
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=50;"
>50% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=75;"
>75% Volume

< INPUT TYPE="radio" NAME="ID_RadioButtonGruppe"
      onpropertychange="ID_Media.mute='false'; ID_Media.volume=100;"
>100% Volume
</BODY>
</HTML>

```

.VolumeName Volumenbezeichner des Laufwerkes

Beispiel:

```

var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk_Bezeichnung = DateiSystem.GetDriveName("C:");
var Laufwerk         = DateiSystem.GetDrive(Laufwerk_Bezeichnung);
var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_TotalerPlatz = Laufwerk.TotalSize;
alert(Laufwerk_Bezeichnung + " " + Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalerPlatz);

```

.vspace vertikaler Abstand in Pixel zum Elternobjekt

.wheelDelta liefert Umdrehung und Richtung in der das Mausrad gedreht wurde
Verwendung bei Event onmousewheel
ab IE 6.x
Objekt event
ist Faktor von 120 Grad-Umdrehung
 > 0 so Mausrad vom User weggedreht
 < 0 so Mausrad zum User gedreht

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Drehen()
    { ID_Span.innerText=event.wheelDelta; }
</SCRIPT>
</HEAD>
<BODY>
    <DIV onmousewheel="Drehen()">Maus hierauf setzen und dann Mausrad drehen ...>
        <SPAN ID="ID_Span"></SPAN>
    </DIV>
</BODY>
</HTML>

```

.width Breite des Objektes in Pixel
Integer in Pixels für absolute Breite, >=0
 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel
String Ziffernfolge eines Integer mit nachfolgendem % für Breite als
 Anteil der Breite des Elternobjektes
 z.B. 10%

.width Auflösung des Bildschirmes in Breite, also Anzahl der horizontalen Pixel
per screen Objekt

.width Auflösung des Bildschirmes in Breite, also Anzahl der horizontalen Pixel
Behavior .style.clientCaps

.wrap Wortumbruch der TEXTAREA
siehe textarea Objekt
"soft" Standard
 Wortumbruch aktiv
 im Formular wird nicht gesendet:
 Zeilenumbruch
 Zeilenvorschub
"hard" Wortumbruch aktiv
 im Formular wird gesendet:
 Zeilenumbruch
 Zeilenvorschub



"off" Wortumbruch nicht aktiv

.x X-Koordinate Maus relativ zum Eltern-Objekt ab IE 5.x
 zum Fenster unter IE 5.x
 X-Koordinate ist relativ zum BODY-Element wenn
 Objekt, das das Event auslöst, absolut positioniert ist (z.B. per Style)
 oder Eltern nicht absolut positioniert sind
 Objekt event
 -1 wenn Maus außerhalb des Fensters

.XMLDocument Referenz auf XML-Dokument (XML-DOM)
 Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
  var ZeigerAufXMLDocument = ID_Div.XMLDocument;
</SCRIPT>
</HEAD>
<BODY>
  <DIV ID="ID_Div">
  </DIV>
</BODY>
</HTML>
```

XMLNS Namensraum für Benutzer-Tags zu einem HTML-Tag
 ab IE 5.x
 nur für das Tag HTML möglich
 Namensraum: Präfix des Benutzer-Tags
 oder Uniform Resource Name (URN)

Beispiel 1:

```
<HTML xmlns:Prefix1 xmlns:Prefix2="www.microsoft.com">
```

Beispiel 2:

```
<HTML xmlns:MSIE>
<HEAD>
  <STYLE>
    @media all { MSIE\;clientCaps {behavior:url(#default#clientcaps);} }
  </STYLE>
</HEAD>
<BODY >
  <MSIE:CLIENTCAPS ..... >
</BODY>
```

.XSLDocument Referenz auf den obersten Knoten des XSL-Dokumentes (Style-Sheet-Dokument)
 var Zeiger = document.XSLDocument;

.y Y-Koordinate Maus relativ zum Eltern-Objekt ab IE 5.x
 zum Fenster unter IE 5.x
 Y-Koordinate ist relativ zum BODY-Element wenn
 Objekt, das das Event auslöst, absolut positioniert ist (z.B. per Style)
 oder Eltern nicht absolut positioniert sind
 Objekt event
 -1 wenn Maus außerhalb des Fensters

Methoden

.abs() Absolutbetrag
 siehe Script-Objekt Math

.acos() Arcus Cosinus im Bogenmass
 liefert Wert von 0 bis PI
 siehe Script-Objekt Math

.activeTimeToParentTime() Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline
 der Eltern konvertieren
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML xmlns:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
  function Anzeige()
```




```

        { alert(ID_Media.syncTolerance; }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(t1.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <t:EXCL ID="ID_EXCL"
        CLASS="time_line_klasse"
        DUR="10s"
        AUTOREVERSE="true"
    >
        <DIV ID="ID_Div1"
            CLASS="time_line_klasse"
            BEGIN="1s"
            DUR="3s"
            TIMEACTION="visibility"
        >
            Erste Zeile
        </DIV>
        <DIV ID="ID_Div2"
            CLASS="time_line_klasse"
            BEGIN="4s"
            DUR="3s"
            TIMEACTION="visibility"
        >
            Zweite Zeile
        </DIV>
    </t:EXCL>
    <BR>
    <BUTTON ID="ID_Button1"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onclick="alert(
            'Punkt in der Timeline der Eltern: '
            + ID_Div1.activeTimeToParentTime(
                ID_EXCL.currTimeState.activeTime
            )
        );
        "
        onrepeat="
            'innerText=Erste Zeile activeTimeToParentTime bei '
            + parseInt(ID_EXCL.currTimeState.activeTime)
            + ' Sekunden';
        "
    >
        Erste Zeile activeTimeToParentTime bei 0 Sekunden
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onclick="alert(
            'Punkt in der Timeline der Eltern: '
            + ID_Div1.activeTimeToParentTime(
                ID_EXCL.currTimeState.activeTime
            )
        );
        "
        onrepeat="
            'innerText=Zweite Zeile activeTimeToParentTime bei '
            + parseInt(ID_EXCL.currTimeState.activeTime)
            + ' Sekunden';
        "
    >
        Zweite Zeile activeTimeToParentTime bei 0 Sekunden
    </BUTTON>
</BODY>
</HTML>

```



`.activeTimeToSegmentTime()` Wert der aktiven Timeline des Elementes in den korrespondierenden Wert der Timeline des Segmentes konvertieren
 per Eigenschaft `.isActive` das Element auf Aktivsein prüfen
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
window.onload = Rekursion;    // ohne () kodieren !!

function Rekursion()
{window.setInterval(Anzeige, 100); }

function Anzeige()
{
    ID_Span1.innerHTML = "&nbsp;activeTimeToSegmentTime:&nbsp;";
                        + (ID_Animate.activeTimeToSegmentTime(
                            ID_Div.currTimeState.activeTime
                        )
                        );

    ID_Span2.innerHTML = "&nbsp;segmentTime:&nbsp;";
                        + (ID_Animate.currTimeState.segmentTime);
}
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Span0"
CLASS="time_line_klasse"
DUR="1"
REPEATCOUNT="indefinite"
onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
>
0
</SPAN>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<t:ANIMATEMOTION ID="ID_Animate"
TARGETELEMENT="ID_Div"
TO="250,0"
DUR="3"
AUTOREVERSE="true"
>
</t:ANIMATEMOTION>
<SPAN ID="ID_Span1">
activeTimeToSegmentTime:
</SPAN>
<BR>
<SPAN ID="ID_Span2">
segmentTime:
</SPAN>
</BODY>
</HTML>
```

`.add()` ein bereits erzeugtes Element einer Collection hinzufügen
 hinzufügen erst nach dem kompletten Laden des Dokumentes
 Element erzeugen per Methode `.createElement()`

Beispiel:

```
<SCRIPT>
var ZeigerAufNeueOption = document.createElement("OPTION");
ID_Select.options.add(ZeigerAufNeueOption);
ZeigerAufNeueOption.innerText = "Option 2";
ZeigerAufNeueOption.Value = "2";
</SCRIPT>
<SELECT ID="ID_Select" >
<OPTION VALUE="1">Option 1</OPTION>
</SELECT>
```



.Add() Date zum Dictionary Objekt hinzufügen

Beispiel:

```
var Datenspeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return Datenspeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    Datenspeicher.add (DatenSchluessel, Date);

    // und Meldung ob Date vorhanden ist
    alert(SchluesselVorhanden(DatenSchluessel));
}
```

.Add() Ordner der Collection Collection FileSystemObject.Folder.Folders hinzufügen
Ordner darf in der Collection nicht bereits existieren (sonst Fehler erzeugt)

Beispiel:

```
var OrdnerName = "c:\\windows\\desktop\\";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder(OrdnerName);
var DateiSystem_FoldersCollection = Ordner.SubFolders;
var NeuerOrdner = DateiSystem_FoldersCollection.Add("Neuer Ordner");
```

.addBehavior() DHTML-Verhaltenseigenschaft einem Element hinzufügen
Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig).
ab IE 5.x bis unter IE 5.5

Beispiel:

```
<SCRIPT>
var FeldDerEigenschaftenID = new Array(); // für removeBehavior
var FeldDerTagsLlimDokument = new Array ();
var FeldDerTagsLlimDokument_Laenge = 0;

function EigenschaftHinzufuegen()
{
    FeldDerTagsLlimDokument = document.all.tags ("LI");
    FeldDerTagsLlimDokument_Laenge = FeldDerTagsLlimDokument.length;
    for (var i=0; i < FeldDerTagsLlimDokument_Laenge; i++)
    {
        var EigenschaftenID // immer neu anlegen wegen Zeigerprüfung
        = FeldDerTagsLlimDokument [i].addBehavior ("hilite.htc");

        if (iEigenschaftenID)
        {FeldDerEigenschaftenID[i] = EigenschaftenID;}
    }
}

function EigenschaftEntfernen()
{
    for (var i=0; i < FeldDerTagsLlimDokument_Laenge; i++)
    {FeldDerEigenschaftenID[i].removeBehavior (FeldDerEigenschaftenID[i]); }
}
</SCRIPT>
<A HREF="javascript:EigenschaftHinzufuegen()">Eigenschaft hinzufuegen</A>
<A HREF="javascript:EigenschaftEntfernen()">Eigenschaft entfernen</A>
```

.AddChannel() Dialogbox zur Channel-Einstellung öffnen um einen Channel zu installieren (Microsoft Active Channel)
erzeugt im Fehlerfall eine Dialogbox und das Event onerror
siehe Objekt window.external

Beispiel:

```
window.external.AddChannel("http://test/file.cdf");
```

.addComponentRequest() Komponente zum Download anfordern und nach dem Download installieren
Behavior .style.clientCaps

.AddDesktop() eine Webseite oder Bild zum installierten Microsoft Active Desktop hinzufügen



wenn Active Desktop nicht installiert, so passiert nichts
siehe Objekt window.external

Beispiel:

```
window.external.AddDesktopComponent("http://www.test.de", "website", 100, 100, 200, 200);
```

.AddFavorite()

Dialogbox zum Hinzufügen einer Url in die Favoritenliste für den User öffnen
siehe Objekt window.external

Beispiel:

```
window.external.AddFavorite(location.href, document.title);
```

.addImport()

StyleSheet aus externer CSS-Datei importieren und als Element der Collection styleSheet.imports erzeugen (entspricht @import url() innerhalb STYLE im HEAD)
siehe Objekt styleSheet und Collection styleSheet.imports

.addPageRule()

StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.pages erzeugen (entspricht @page vom Objekt page)
siehe Objekt styleSheet und Collection styleSheet.pages

Beispiel:

```
function TextFaerben ()
{document.styleSheets[0].addPageRule("DIV B", "color:blue", 0);}

<DIV onmouseover="TextFaerben ();">
  <B>dieser Text wird per CSS mit blauer Fabrbe angezeigt</B>
</DIV>
```

.addReadRequest()

Eintrag in Queue für Lesezugriff (Request Queue) erzeugen
siehe Objekt navigator.userProfile und Autovervollständigung im Internet Explorer

Beispiel :

```
// Request Queue füllen

//      es soll der Wert zu "vcard.displayname" ermittelt werden
navigator.userProfile.addReadRequest("vcard.displayname");
//      es soll der Wert zu " vcard.gender " ermittelt werden
navigator.userProfile.addReadRequest("vcard.gender");

// Datenermitteln per Queue
navigator.userProfile.doReadRequest(12, "Daten von unbekannt verwendet");

// Queue leeren
navigator.userProfile.clearRequest();
```

.addRule()

StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.rules erzeugen (entspricht xx { ...} innerhalb STYLE im HEAD)
siehe Objekt styleSheet und Collection styleSheet.rules

Beispiel:

```
function TextFaerben ()
{document.styleSheets[0].addRule("DIV B", "color:blue", 0);}

<DIV onmouseover="TextFaerben ();">
  <B>dieser Text wird per CSS mit blauer Fabrbe angezeigt</B>
</DIV>
```

.alert()

Dialogbox erzeugen

1. Anzeige von:
 - Ausrufungszeichen-Symbol
 - variablen String
 - OK-Button und
 - warten auf Drücken des OK-Button
- 2.

siehe Objekt window

.anchor()

HTML-Anker <A> als HTML-Kette erzeugen in der Form "text"
und ohne weitere Attribute ID, HREF etc.
siehe Script-Objekt String

Beispiel:

```
var StringLiteral    ="Sichtbarer Ankertext"
var HTML_Kette      = StringLiteral.anchor("WertDesNAMEAttributes");
HTML_Kette          = "Sichtbarer Ankertext".anchor("WertDesNAMEAttributes");

entspricht <A NAME="WertDesNAMEAttributes">Sichtbarer Ankertext</A>
```

```
eval(HTML_Kette);           erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette); zeigt den Anker an und erzeugt Eintrag in der Collection document.anchors
```



.appendChild() Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern
 DOM wird geändert
 Zeiger wird zugleich immer am Ende der Collection `childNodes` angehängen
 Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde

Beispiel 1:

```
var ZeigerAufDiv = document.createElement("DIV");
document.body.appendChild(ZeigerAufDiv);
```

Beispiel 2:

```
<SCRIPT>
function Anhaengen()
{
    var ZeigerAufNeuesLI oNewNode = document.createElement("LI");
    Liste.appendChild(ZeigerAufNeuesLI);           // dem BODY anhängen
                                                    // also sichtbar werdend
    ZeigerAufNeuesLI.innerHTML="Listenelement 5";
}
</SCRIPT>
<BODY>
    <UL ID = "Liste" >
        <LI>Listenelement 1
        <LI>Listenelement 2
        <LI>Listenelement 3
        <LI>Listenelement 4
    </UL>
    <INPUT TYPE = "button" VALUE = "Anhaengen "
           onclick = " Anhaengen()">
</BODY>
```

.appendData() String an das Ende des Objektes anhängen

.apply() ein anderes Objekt anstelle des arguments Script-Objekt verwenden
 siehe Script-Objekt Funktion und Script-Objekt arguments und Anweisung function

.applyElement() Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern
 DOM wird geändert
 Element kann selbst Kinder haben
 Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde
 Achtung: Wenn Element per Methode `.createElement()` erzeugt wurde, aber nicht bereits im Dokumentenbaum eingebunden ist, so wird die Eigenschaft `.innerHTML` gelöscht !

Beispiel:

```
<SCRIPT>
function Hinzufuegen()
{
    var Zeiger = document.createElement("LI");
    Liste.applyElement(Zeiger);
}
</SCRIPT>
<UL ID = Liste>
    <LI>Listenelement 1
    <LI>Listenelement 2
    <LI>Listenelement 3
    <LI>Listenelement 4
</UL>
<INPUT TYPE="button" VALUE=" Hinzufuegen" onclick=" Hinzufuegen()">
```

.asin() Arcus Sinus im Bogenmass
 liefert Wert von $-\pi/2$ bis $+\pi/2$
 siehe Script-Objekt Math

.assign() neues Dokument zuweisen und laden
 im Verlauf (History) wird ein neuer Eintrag hinzugefügt (history Objekt)
 Altes Dokument ist per Vorwärts- und Zurück-Button einstellbar.

.atan() Arcus Tangens im Bogenmass
 liefert Wert von $-\pi/2$ bis $+\pi/2$
 siehe Script-Objekt Math

.atan2() Arcus Tangens im Bogenmass
 siehe Script-Objekt Math

.atEnd() prüfen auf Erreichen des Ende der Collection



auf Zustand undefined eines Elementes der Collection
 auf leere Collection
 verwenden mit .moveNext() innerhalb einer Schleife
 siehe Enumerator JScript-Objekt

Beispiel Laufwerke auf dem PC des Users ermitteln:

```
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd(); DateiSystem_Laufwerke.moveNext() )
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);
```

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
 Hinweis: Abschalten mit Methode .detachEvent()
 Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
 NICHT verkettet sondern in **Zufallsfolge**, es sei denn
 die Handler prüfen ihre Aufruffolge (muss programmiert werden)
 true Einschalten erfolgreich
 false Einschalten nicht möglich gewesen

Beispiel:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
function CursorNeu()
{
    if (event.srcElement == element)
    {
        normalColor      = style.color;
        runtimeStyle.color = "red";
        runtimeStyle.cursor = "hand";
    }
}

function CursorNormal()
{

```



```

        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu);           // nicht () kodieren !!!
        detachEvent ('onmouseout', CursorNormal);         // nicht () kodieren !!
    }

    attachEvent ('onmouseover', CursorNeu);
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```

.attachEvent()

Einschalten des Registrieren eines Events durch Eventhandler

Hinweis: Abschalten mit Methode .detachEvent()

Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider

NICHT verkettet sondern in **Zufallsfolge**, es sei denn

die Handler prüfen ihre Aufruffolge (muss programmiert werden)

Vor dem Neuelegen immer den Standard-Eventhandler retten und diesen

nach .detachEvent() wieder einbinden (siehe Beispiel).

siehe Objekt window

Beispiel 1:

```

<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
        window.onmouseover = Rette_Standard_Eventhandler_OnMouseOver;

        detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
        window.onmouseout = Rette_Standard_Eventhandler_OnMouseOut;
    }

    var Rette_Standard_Eventhandler_OnMouseOver = window.onmouseover;
    attachEvent ('onmouseover', CursorNeu);

    var Rette_Standard_Eventhandler_OnMouseOut = window.onmouseout;
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```

Beispiel 2:

```

function ResizeHandler()           // Homepage neu laden
{ window.history.go(0); }

// Standardhandler retten
var RetteHandler_Resize = window.onresize;

// und neu belegen
window.onresize = ResizeHandler;   // Homepage neu laden
                                   // ohne () kodieren, damit nicht sofort ausgeführt wird

```




```
// und Start des Abfangens vom resize-Event
window.attachEvent('onresize', ResizeHandler);
```

```
.....
```

```
// irgendwann abschalten der Eventüberwachung
window.detachEvent('onresize', ResizeHandler);
```

```
// und Standardhandler wieder einbinden
window.onresize = RetteHandler_Resize;
```

`.AutoCompleteSaveForm()` permanentes Speichern von Daten eines Formulars in den AutoComplete-Data Store
Diese Methode ist mit Vorsicht zu geniessen und richtet sich nach dem aktuellen Einstellungen von AUTOCOMPLETE, die der User in den Browser-Optionen treffen kann.
 Gespeichert werden die Werte von INPUT's im Formular, die das Attribut NAME besitzen, wobei auch INPUT TYPE=password speicherbar ist
 Hinweis: Im Formular-Tag ist ebenfalls NAME zu kodieren
 NAME-Attribut ist auch Voraussetzung für das Senden von Formulardaten an den Server der Webseite
 AutoComplete-Data Store kann vom User gelöscht werden (auch komponentenweise).

Beispiel:

```
<SCRIPT>
function Speichern()
{
    // erst speichern
    window.external.AutoCompleteSaveForm(ID_Formular);
    // dann löschen
    ID_Formular.ID_Input1.value="";
    ID_Formular.ID_Input2.value="";
}
</SCRIPT>
<FORM NAME="ID_Formular">
  Dieser Text wird gespeichert:
  <INPUT TYPE="text" NAME="ID_Input1">
  Dieser Text wird nicht gespeichert:
  <INPUT TYPE="text" NAME="ID_Input2" AUTOCOMPLETE="off">
</FORM>
<INPUT TYPE=button VALUE="Speichern" onclick=" Speichern()">
```

`.AutoScan()` Url einer beliebigen Webseite festlegen, die als Fehlermeldung vom Autoscan angezeigt wird
 Webseite wird aufgerufen, wenn Autoscan nicht erfolgreich war
 erklärt den Misserfolg
muss immer erreichbar, also anzeigbar sein
 Standardwebseite auf Festplatte des User-PC vorhanden (falls der User diese nicht manuell gelöscht hat)
 Autoscan: Suchen nach einer anzuzeigenden Web-Seite im Internet durch den Browser nach Eingabe der Url (auch mit Autovervollständigung)
 Voraussetzungen für Autoscan:
 es muss aktiv sein
 Autoscan leider nur möglich für Domains mit
 "www" am Anfang der Url
 Suffixe der Url laut Registry-Eintrag
 HKEY_LOCAL_MACHINE\software\microsoft\internet explorer\main\urltemplate
 (standardgemäß steht dort .com, .org, .net, und .edu)
 Suche in der Reihenfolge der Einträge laut dem Registry-Eintrag
 Hinweis: Der Registry-Eintrag ist manuell änderbar:
 Bsp.: .de als Eintrag hinzufügen an gewünschte Position
 siehe Objekt window.external

Beispiel:

```
window.external.AutoScan("test","error.htm","_main");
                                für www.test.xxx                                mit xxx als Domain-Suffix laut Registry-Eintrag
                                                                (siehe oben)
```

`.back()` Aktivierung einer zur aktuell besuchten Seite im Verlauf des Surfens davorliegenden Seite

`.beginElement()` Element auf der Timeline starten, also aktivieren
 identisch mit Wirkung, wenn Startzeit des Elementes erreicht wird auf der Timeline
 Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes.
 per Eigenschaft `.isActive` das Element auf Aktivsein prüfen
 siehe Objekt `currTimeState` und `Behavior.style.time2`
 -1 entspricht wie wenn nicht kodiert



entspricht Druck des Back-Buttons
 je kleiner umso weiter zuvorliegend
 entspricht Selektion aus der Verlaufsliste

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SEQ ID="ID_Seq"
CLASS="time_line_klasse"
>
<SPAN ID="ID_Span1"
CLASS="time_line_klasse"
TIMEACTION="display"
DUR="1"
>
5...
</SPAN>
<SPAN ID="ID_Span2"
CLASS="time_line_klasse"
TIMEACTION="display"
DUR="1"
>
4...
</SPAN>
<BR>
<IMG ID="ID_Img"
SRC="test.gif"
CLASS="time_line_klasse"
DUR="indefinite"
>
</t:SEQ>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Seq.beginElement();">
Start der Timeline
</BUTTON>
</BODY>
</HTML>
```

.beginElementAt() Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren)
 also mit Wartezeit ab Beginn der Timeline des Elementes
 wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so startet das Kindelement sofort,
 also ohne Wartezeit
 wenn zusätzlich ein weiterer Start (oder mehrere Starts)
 mit identischem oder kleinerem Zeitpunkt kodiert, so wird dieser zusätzliche ignoriert
 mit größerem Zeitpunkt kodiert, so wird dieser Start auch ausgeführt
 also Neustart nach einem erfolgten Start
 per Eigenschaft **.isActive** das Element auf Aktivsein prüfen
 siehe Objekt **currTimeState** und Behavior **.style.time2**

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function Starten()
{
// prüfen ob Elternobjekt, also body, den Startzeitpunkt schon erreicht hat
// bzw. der Startwert falsch ist
if ( (ID_Input.value <= document.body.currTimeState.activeTime)
|| (ID_Input.value==null)
)
{
alert('Startwert falsch');
ID_Input.value="";
ID_Input.focus();
return;
}
}
```



```

        else
        { ID_Excl.beginElementAt(ID_Input.value); }
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <SPAN ID="ID_Span2"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Excl.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <t:EXCL ID="ID_Excl"
        BEGIN="0"
        CLASS="time_line_klasse"
        DUR="27"
    >
        <DIV ID="ID_Div1"
            CLASS="time_line_klasse"
            BEGIN="1"
            DUR="5"
        >
            Erste Zeile
        </DIV>
        <DIV ID="ID_Div2"
            CLASS="time_line_klasse"
            BEGIN="6"
            DUR="5"
        >
            Zweite Zeile
        </DIV>
    </t:EXCL>
    <INPUT type="text" ID="ID_Input" SIZE="3" >
    <BUTTON ID="ID_Button" onclick="Starten()">
        Start
    </BUTTON>
</BODY>
</HTML>

```

.big() HTML-Tag <BIG> erzeugen in der Form <BIG>text</BIG>
 siehe Script-Objekt String

Beispiel:

```

var StringLiteral    ="Text der BIG wird"
var HTML_Kette      = StringLiteral.big();
HTML_Kette           = "Text der BIG wird".big();

```

entspricht <BIG>Text der BIG wird</BIG>

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
 document.write(HTML_Kette);

.blink() HTML-Tag <BLINK> erzeugen in der Form <BLINK>text</BLINK>
 siehe Script-Objekt String

Beispiel:

```

var StringLiteral    ="Text der BLINK wird"
var HTML_Kette      = StringLiteral.blink();
HTML_Kette           = "Text der BLINK wird".blink();

```

entspricht <BLINK>Text der BLINK wird</BLINK>



```
eval(HTML_Kette);           erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

.blur() Element den Focus wegnehmen und Event onblur auslösen
Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
vor IE 5.0 TABINDEX-Attribut muss kodiert sein
ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein

.blur() Element den Focus wegnehmen und Event onblur auslösen
Der Focus wird **nicht automatisch** auf irgend ein anderes Element gesetzt !
vor IE 5.0 TABINDEX-Attribut muss kodiert sein
ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
siehe Objekt window

.bold() HTML-Tag erzeugen in der Form text
siehe Script-Objekt String

Beispiel:

```
var StringLiteral    ="Text der BOLD wird"
var HTML_Kette       = StringLiteral.bold();
HTML_Kette           = "Text der BOLD wird".bold();
```

entspricht Text der BOLD wird

```
eval(HTML_Kette);           erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

Boolean() konvertiert einen Wert nach Boolean

.BuildPath() PATH-Variable im Dateisystem erweitern durch Anhängen
keine Erweiterung des Dateisystems per Ordner (dafür gibt es andere Methoden)

.call() eine Methode eines anderen Objektes aufrufen: Argumentenliste beachten
siehe Script-Objekt Funktion und Script-Objekt arguments und Anweisung function

.ceil() nächste ganze Zahl oberhalb zahl ermitteln
Bsp: ceil(1.1) ergibt 2
 ceil(1) ergibt 1
siehe Script-Objekt Math

.charAt() Zeichen liefern unter Nutzung des Indexes
siehe Script-Objekt String

Beispiel 1:

```
var StringLiteral="Text"
StringLiteral.charAt(0)     liefert "T"
StringLiteral.charAt(5)     liefert Leerkette
"Text".charAt(0)           liefert "T"
```

Beispiel 2 Zeichenkette auf Wertebereich der Zeichen prüfen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function pruefe_zeichenkette(zeichenkette, wertebereich)
{
    // wertebereich ist Zeichenkette z.B. "0123456789 -+/,()"

    var rueckgabewert = true;           // Annahme: Zeichenkette hat NUR Zeichen
                                         // aus dem Wertebereich

    var zeichen_aus_zeichenkette;

    // Zeichenkette zeichenweise analysieren: Jedes Zeichen mit Wertebereich vergleichen
    for (var i = 0; i < zeichenkette.length; i++)
    {
        zeichen_aus_zeichenkette = zeichenkette.charAt(i);

        if (wertebereich.indexOf(zeichen_aus_zeichenkette) == -1)
        {
            rueckgabewert = false;
            break;           // ungültiges Zeichen gefunden, also abbrechen
        }
    }

    return rueckgabewert;
}
```



```

function pruefe_eingabe(zu_pruefende_zeichenkette)
{
    if (pruefe_zeichenkette(zu_pruefender_zeichenkette, "0123456789 -+/.()"))
    {alert("Eingabe ist korrekt !");}
    else
    {alert("Eingabe ist nicht korrekt !"); }
}

//-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    Telefon:
    <INPUT TYPE="text"
        NAME="Telefon"
        VALUE=""
    >
    <INPUT TYPE="button"
        VALUE="Ueberpruefen"
        onclick="pruefe_eingabe(this.form.Telefon.value)">
</FORM>
</BODY>
</HTML>

```

.charCodeAt() Unicode des Zeichen liefern unter Nutzung des Indexes
Unicode von 0 bis 65535, wobei
0 bis 127 der ASCII-Zeichensatz ist
0 bis 255 der ISO-Latin-1-Zeichensatz ist
siehe Script-Objekt String

Beispiel:

```

var StringLiteral ="Text";
StringLiteral.charCodeAt(0)    liefert 84 für "T"
StringLiteral.charCodeAt(5)    liefert NaN
"Text".charCodeAt(0)           liefert 84 für "T"

```

.clear() Dokument löschen
nur Objekt document

.clear() Selektion als Markierung aufheben per document.selection Objekt des Dokumentes per document.selection
nicht Selektion löschen
siehe Methode .empty()

.clearAttributes() alle HTML-Attribute eines Objektes entfernen
außer ID, STYLE und per Script definierte Attribute
Script-erzeugte Attribute nicht entfernenbar
DOM wird geändert

Beispiel:

```

<SCRIPT>
function Loeschen()
{
    ID_Span.children[0].clearAttributes();
}
</SCRIPT>
<SPAN ID="ID_Span">
    <DIV ID="ID_Div"
        ATTRIBUTE1="true"
        ATTRIBUTE2="true"
        onclick="alert('click');"
        onmouseover="this.style.color='#0000FF';"
        onmouseout="this.style.color='#000000';"
    >
        Test eines<B>Div</B>Elementes.
    </DIV>
</SPAN>
<INPUT TYPE="button" VALUE=" Loeschen" onclick="Loeschen()">

```

.clearComponentRequest() Puffer der per .addComponentRequest() angeforderten Downloads löschen
Behavior .style.clientCaps

.clearData() Clipboardinhalt löschen
Anwendung für Ereignisse ondragstart oder ondrop
"Text" zu löschende Daten sind Text-Format
"URL" zu löschende Daten sind im URL-Format.



"File" zu löschende Daten sind im File-Format.
 "HTML" zu löschende Daten sind im HTML-Format
 "Image" zu löschende Daten sind im Image-Format
 Leerkette
 wenn **nichts** kodiert, so werden **alle** Datenformat im Clipboard **gelöscht**

.clearInterval() stoppt einen Timer, der mit .setInterval() gestartet wurde
 siehe Objekt window

Beispiel 1:

```
var timerID = null;

function timer()
{
    // tue was
}

function starttimer()
{
    if (timerID == null)
    {timerID = setInterval("timer()", 1000);}
}

function stoptimer()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
    }
}
```

Beispiel 2 für Sekundenbalken:

```
<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }
}
```



```

    }
}

function Stoppen()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
        ID_Button1.disabled = false;
        ID_Button2.disabled = true;
    }
}

function ZurueckSetzen()
{ Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

.clearRequest() Queue für Lesezugriff (Request Queue) leeren
siehe Objekt navigator.userProfile

Beispiel :

```

// Request Queue füllen

// es soll der Wert zu "vcard.displayname" ermittelt werden
navigator.userProfile.addReadRequest("vcard.displayname");
// es soll der Wert zu " vcard.gender " ermittelt werden
navigator.userProfile.addReadRequest("vcard.gender");

// Datenermitteln per Queue
navigator.userProfile.doReadRequest(12, "Daten von unbekannt verwendet");

// Queue leeren
navigator.userProfile.clearRequest();

```

.clearTimeout() löscht ein Timeout, das mit .setTimeout() gesetzt wurde
siehe Objekt window

.click() simuliert einen Klick auf das Element und löst onclick-Event aus
manipuliert nicht den Focus

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function ClickMitFocus()
    {

```




```

        ID_CheckBox.focus();
        ClickOhneFocus();
    }

    function ClickOhneFocus ()
    { ID_CheckBox.click();}
</SCRIPT>
<SCRIPT FOR= ID_CheckBox EVENT=onfocus>
    alert("Check Box hat Focus");
</SCRIPT>
</HEAD>
<BODY>
    <INPUT Type="CHECKBOX" ID="ID_CheckBox"></INPUT>
    <BR>
    <BUTTON onclick="ClickMitFocus()">Klick mit Fokus</BUTTON>
    <BR>
    <BUTTON onclick="ClickOhneFocus()">Klick ohne Fokus</BUTTON>
</BODY>
</HTML>

```

.cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
 DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)

Beispiel:

```

<SCRIPT>
    function Klonen()
    {
        var ZeigerAufKlone = Liste.cloneNode(true); // DOM wird nicht geändert
        document.body.insertBefore(ZeigerAufKlone); // DOM wird geändert
    }
</SCRIPT>
<UL ID = "Liste" >
    <LI>Listenelement 1
    <LI>Listenelement 2
    <LI>Listenelement 3
    <LI>Listenelement 4
</UL>
<INPUT TYPE="button" VALUE=" Klonen" onclick=" Klonen()">

```

.close() Ausgabe-Datenstream schliessen und Anzeige des Dokumentes beenden
 schliessen einer mit .open() erzeugten Dokument-Instanz
 Objekt document

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function ErzeugeZurLaufZeit()
    {
        // Dokumentinstanz erzeugen
        var ID_Dokument = document.open("text/html", "replace");

        // Dokumentinhalt festlegen und anzeigen
        ID_Dokument.write("<HTML><BODY>Hallo</BODY></HTML>");

        // Dokumentinstanz schliessen
        ID_Dokument.close();
    }
</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE="button" onclick=" ErzeugeZurLaufZeit();">
</BODY>
</HTML>

```

Beispiel:

```

function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");

```



```
var FensterDokumentZeiger = FensterZeiger.document;
```

```
var Kette= '<HTML>'
          + '<HEAD></HEAD>'
          + '<BODY >'
          + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
          + '<BR>'
          + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
          + ' onclick="opener.OnClickHandler();"'
          + '>'
          + '</BODY>'
          + '</HTML>';
```

```
FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.close() Fenster schliessen, das offen ist
 Fenster muss nicht explizit mit Methode .open()Methode erzeugt worden sein
 Schliessen des letzten Browserfensters erzeugt immer Dialog-Box-Abfrage
 Fenster des Dokumentes schliessen: document.close()
 innerhalb von Eventhandler: Immer **window.close()** oder logischer_window_name.close() oder self.close()
 kodieren
 siehe Objekt window

Beispiel:

```
<BODY onclick="window.close();">
  Klick zum Schliessen des Fensters
</BODY>
```

Beispiel für Fenster schliesst sich selbst nach Wartezeit:

```
</HTML>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function fenster_offnen_und_schliessen()
    {

        var fenster;
        fenster=window.open("", "Fenster", "width=180,height=100");
        fenster.document.write("<H1>Ich schließe mich nach 4 Sekunden</H1>");
        fenster.setTimeout('window.close()',4000);

    }

-->
</SCRIPT>
</HEAD>
<BODY onload="fenster_offnen_und_schliessen()">
</BODY>
</HTML>
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
          + '<HEAD></HEAD>'
          + '<BODY >'
          + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
          + '<BR>'
          + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
          + ' onclick="opener.OnClickHandler();"'
          + '>'
          + '</BODY>'
          + '</HTML>';
```



```
FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.Close() offene Datei schliessen
 nach Schliessen der Datei sind keine Methoden mehr verwendbar, ohne die Datei vorher neu zu öffnen

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("Test");
DateiOffen.Close();
```

.collapse() Textbereich-Zeichen-Zeiger auf Anfang oder Ende des Textbereiches setzen
 Hinweis: Zeiger nur auf Plain-Text-Zeichen
 Bsp.: <BODY><P>abc
 Zeiger kann wandern von VOR a bis HINTER c
 VOR a entspricht Start des Bereiches
 mit Zeigerwert 0
 HINTER c entspricht Ende des Bereiches
 mit Zeigerwert 3

per textrange Objekt
 nur unter Windows 32-Bit

Beispiel:

```
<SCRIPT>
function Selektion()
{
    var ZeigerAufSelektionsQuelle = window.event.srcElement ;

    if (!ZeigerAufSelektionsQuelle.isTextEdit)
    { ZeigerAufSelektionsQuelle = ZeigerAufSelektionsQuelle.parentTextEdit;}

    if (ZeigerAufSelektionsQuelle != null)
    {
        var ZeigerAufTextBereich =
            ZeigerAufSelektionsQuelle.createTextRange();

        ZeigerAufTextBereich.moveToElementText(window.event.srcElement);
        ZeigerAufTextBereich.collapse();
        ZeigerAufTextBereich.expand("SelektionsText");
        ZeigerAufTextBereich.select();
    }
}
</SCRIPT>
```

.compareEndpoints() Vergleich der Textbereich-Zeichen-Zeiger von 2 Textbereichen
 Hinweis: Zeiger nur auf Plain-Text-Zeichen
 Bsp.: <BODY><P>abc
 Zeiger kann wandern von VOR a bis HINTER c
 VOR a entspricht Start des Bereiches
 mit Zeigerwert 0
 HINTER c entspricht Ende des Bereiches
 mit Zeigerwert 3

per textrange Objekt
 nur unter Windows 32-Bit

.compareVersions() 2 Versionen einer Komponente vergleichen
 Bsp. für Version "5,0,18,1024"
 Behavior .style.clientCaps

.componentFromPoint() Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
 auch für CSS-Layout
 onmouseover-Event hat **nicht die Pixelgenauigkeit** wie die Angaben der Methode .componentFromPoint()
 also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos
 erreicht haben
 Overbereich der Maus ist mehr als 1 Pixel gross
 beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function ComponentSnapShot()
{
    var LayoutKomponente =
        document.body.componentFromPoint( event.clientX,
```



```

        event.clientY
    );

    if (LayoutKomponente == "")
    { alert("Keine Layout-Komponente"); }
    else
    { alert("Layout-Komponenten = " + LayoutKomponente); }
}

function trackElement()
{
    var LayoutKomponente =
        document.body.componentFromPoint( event.clientX,
                                           event.clientY
                                           );

    window.status =
        "mousemove "
        + event.clientX
        + ", "
        + event.clientY
        + " Layout-Komponente = "
        + LayoutKomponente;
}
</SCRIPT>
</HEAD>
<BODY onmousemove="trackElement()"
      onmousedown="ComponentSnapShot()"
      onkeydown="ComponentSnapShot()"
      oncontextmenu="ComponentSnapShot()"
>
    <TEXTAREA COLS=500 ROWS=500>
        Dieser Text erzeugt Scrollbalken.
    </TEXTAREA>
</BODY>
</HTML>

```

`.concat()` Feld (Quellfeld1) kopieren in eine neue und automatisch erzeugte Instanz (Zielfeld) und optionales Anhängen von Werten (z.B. weiteren Quellfeldern2 ...) an das Ende des Zielfeldes. Quellfeld1 kann leer sein
Verkettung erfolgt in der Folge der Kodierung der zu verkettenden Objekte, Ausdrücke etc.:
Erstes Objekt in der Verkettung ist immer Quellfeld1.
wenn mehrere Quellfelder zu verketteten sind, dann gilt
wenn Feldelement **nicht** String **und nicht** Number ist, so
wird das Feldelemente als Zeiger kopiert bzw. verkettet
der Wert, auf den der Zeiger weist, wird nicht verwendet
bewirkt Änderung eines Wertes im Quell- bzw. Zielfeld auch die Änderung im Ziel- bzw. Quellfeld, da identische Zeigerbezüge vorliegen
wenn Feldelement String oder Number ist, so
erfolgt Wertkopierung, also keine Zeigerkopierung
bewirkt Änderung eines Wertes im Quell- bzw. Zielfeld keine Änderung im Ziel- bzw. Quellfeld, da keine Zeigerbezüge vorliegen
Änderung der Anzahl der Feldelemente in den Quellfeldern hat keine Auswirkung auf die Anzahl der Feldelemente im Zielfeld
Verkettung von Feldern erfolgt
feldweise in der Folge der Kodierung der Quellfelder
und pro Quellfeld: elementweise in der Folge der Feldelemente
Felder haben die Objektklasse Script-Objekt Array
ab IE 5.5 und NS 6.x siehe auch `.push()`

Beispiel:

```

var Feld1      = new Array(0,1);
var Feld2_Quelle = new Array(2, 3);
var Wert_Quelle = 4;
var Feld3_Ziel = Feld1.concat(Feld2_Quelle, Wert_Quelle,);
alert(Feld3.join()); // "0,1,2,3,4"

```

Beispiel:

```

var QuellFeld1=new Array("a","b","c");
var QuellFeld2=new Array(1,2,3);
var ZielFeld=QuellFeld1.concat(QuellFeld2) // Zielfeld enthält Wertkopien also ["a","b","c",1,2,3]

```

Beispiel:

```

var Variable1="Hallo ";
var Variable2="Du";
var Variable3="!";

var QuellFeld1=new Array(Variable1,Variable2,Variable3);

```




```

<SCRIPT>
    function Hinzufuegen()
    {
        // Attribut mit Wert erzeugen
        var ZeigerAufAttribut = document.createAttribute("title");
        ZeigerAufAttribut.value = "Tooltip-Text ";

        // Attribut als Feldelement anhängen
        var ZeigerAufFeld = ID_Div.attributes;
        ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
    }
</SCRIPT>
</HEAD>
<BODY onload=" Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

`.createCaption()` leeres CAPTION in einer Tabelle erzeugen **und** einbinden
es darf nur 1 CAPTION zur Tabelle existieren
siehe Objekt table

`.createComment()` Comment-Objekt (Kommentar-Objekt) im Dokument erzeugen und Referenz liefern
verwendbar anstelle von direkt im HTML- bzw. Script-Quellcode kodiertem Kommentar
DOM wird geändert, da das Dokument erweitert wird

Beispiel:

```
var Kommentar = document.createComment("Das ist ein Kommentar");
```

`.createControlRange()` Selektionsbereich erzeugen
es kann nur genau 1 Range pro Zeitpunkt existieren; wenn einer vorhanden, so diesen überschreiben

Beispiel 1:

```
var Zeiger = document.body.createControlRange();
```

Beispiel 2:

```

function ControlRangeErzeugenUndSelektieren()
{
    var ControlRangeObjekt = document.body.createControlRange();
    ControlRangeObjekt.add(document.all.zeiger_auf_control_element);
    ControlRangeObjekt.select();
}

```

`.createDocumentFragment()` erzeugt neues Dokument
Objekt document

`.createElement()` HTML-Objekt (Tags) im Dokument erzeugen und Referenz liefern
Achtung: Erzeugtes Objekt muss in DOM noch per Methode `.insertBefore()` bzw. `.appendChild()` eingereiht werden.
Hinweis: Attribute mit der Methode `.createAttribute()` erzeugen
DOM wird geändert

Beispiel 1:

```

<SCRIPT>
    function Erzeugen()
    {
        ID_Span.innerHTML="";
        var FeldDerZeigerAufOption = ID_Select.options[ID_Select.selectedIndex];

        if(FeldDerZeigerAufOption.text.length>0)
        {
            var ZeigerAufElement=
                document.createElement(FeldDerZeigerAufOption.text);
            eval(
                " ZeigerAufElement."
                + FeldDerZeigerAufOption.value
                + "="
                + ID_Input.value
                + ""
            );

            if(FeldDerZeigerAufOption.text=="A")
            { ZeigerAufElement.href="javascript:alert('A link.');" }

            ID_Span.appendChild(ZeigerAufElement);
        }
    }
</SCRIPT>
<SELECT ID="ID_Select" onchange="Erzeugen()">

```



```

        <OPTION VALUE="innerText">A
        <OPTION VALUE="value">&lt;INPUT TYPE="button"&gt;
    </SELECT>
    <INPUT TYPE="text" ID="ID_Input" VALUE="Beispiel Text">
    <SPAN ID="ID_Span" ></SPAN>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Erzeuge()
    {
        var Zeiger = document.createElement(
            "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Eins'>"
        );
        document.body.insertBefore(Zeiger);

        Zeiger = document.createElement(
            "<INPUT TYPE='RADIO' NAME='RADIOTEST' VALUE='Zwei'>"
        );
        document.body.insertBefore(Zeiger);
    }
</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE="BUTTON"
        ONCLICK=" Erzeuge()"
        VALUE="Zwei Radio Buttons erzeugen">

    <BR>
    <INPUT TYPE="BUTTON"
        ONCLICK="alert(document.body.outerHTML)"
        VALUE="Click um HTML zu sehen">

    <BODY>
</HTML>

```

.createEventObject() Event-Objekt im Dokument erzeugen nur für Methode .fireEvent()

Beispiel:

```

var EventObjekt = document.createEventObject();
document.fireEvent("onclick", EventObjekt);

```

.createEventObject() Event erzeugen in einem Eventhandler einer HTC-Komponente
siehe Objekt element und HTC-Datei

Beispiel:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
    <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

    <SCRIPT LANGUAGE="JScript">
        function Berechne()// Bezug über ID_Event
        {
            var EventObjekt    = createEventObject();
            EventObjekt.result = "Wert";

            ID_Event.fire (EventObjekt);
        }
    </SCRIPT>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
    @media all { privater_tag_namensraum:privater_tag {behavior:url(test.htc)} }
</STYLE>
</HEAD>
<BODY>
    <privater_tag_namensraum:privater_tag
        ID="ID_privater_tag"
        onResultChange="ID_Div.innerText=window.event.result"
    >

    <TABLE>

```




```

<TR>
  <DIV ID="ID_Div"
    ALIGN=RIGHT
    STYLE="border: '.025cm solid gray'"
  >
    0.
  </DIV>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" % " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" + ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" = ">

```



```

</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

.CreateFolder() Ordner erzeugen, der nicht bereits existieren darf (sonst wird Fehler erzeugt)

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.CreateFolder("c:\\test")

```

.createPopup() Pop-up-Fenster ohne irgendwelche Fensterelemente öffnen z.B. für Anzeige von Tooltips
 siehe Objekt window.popup
 Fenster wird mit der Anzeige per Methode .show() zum aktuellen Fenster
 erst geschlossen, wenn **anderes Fenster** aktuell wird
 z.B. durch Klicken außerhalb des Pop-up-Fensters

ab IE 5.5

siehe Objekt window

Beispiel

```

var PopUpID=window.createPopup();
PopUpID.document.body.style.backgroundColor="green";
PopUpID.document.body.innerHTML="TooltipText";
PopUpID.show(100,100,180,25,document.body);

```

.createRange() Zeiger auf einen Universal-Bereich erzeugen per document.selection Objekt des Dokumentes
 Universal-Bereich kann enthalten:

Text (document.selection.textRange Collection
 textRange Objekt)

oder Control-Element(e) (document.selection.controlRange Collection)

siehe auch Methoden .createControlRange() .createTextRange() und .createRangeCollection()
 Eigenschaft .type

.createRangeCollection() document.selection.textRange Collection erzeugen per document.selection Objekt des Dokumentes
 nur wenn der Browser multiple Selektion unterstützt, so mehr als 1 Feld-Element textRange Objekt
 vorhanden bei Mehrfach-Selektion durch User
 siehe auch textRange Objekt
 Methoden .createRange() .createControlRange() und .createTextRange()

.createStyleSheet() Style-Sheet-Objekt im Dokument erzeugen und Referenz liefern
 DOM wird geändert

Beispiel:

```

document.createStyleSheet('styles.css');

```

.CreateTextFile() Textdatei anlegen und zum Schreiben als Textstream öffnen
 Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.CreateTextFile("c:\\test.txt", true)
DateiOffen.WriteLine("Test");
DateiOffen.Close();

```

.createTextNode() Plain-Textelement im Dokument erzeugen und Referenz liefern
 Plaintext kann keine HTML-Tags enthalten
 DOM wird geändert, da Dokument erweitert wird

Beispiel:

```

<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
Original Text
</SPAN>

```

.createTextRange() Textbereich erzeugen

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
var FeldAllerButtonElemente coll = document.all.tags("BUTTON");

```



```

        if ( (FeldAllerButtonElemente !=null )
            && (FeldAllerButtonElemente.length>0)
        )
        {
            var ZeigerAufRange = FeldAllerButtonElemente[0].createTextRange();
            ZeigerAufRange.text = "Clicked";
        }
    </SCRIPT>

```

Beispiel 2:

```

function ErzeugeUndSelektiereTextRange()
{
    var TextBereich = document.body.createTextRange();
    TextBereich.findText("Testtext");
    TextBereich.select();
}

```

Beispiel 3 Einbindung einer Suchmaschine:

```

var markierter_text=document.selection.createRange().text;
    // oder var markierter_text=prompt('Suchbegriff: ');
var suchmaschinen_url='http:// .....';
var suchmaschinen_parameter='.....';

if (markierter_text)
{location.HREF=suchmaschinen_url + suchmaschinen_parameter + escape(markierter_text);}
else
{location.HREF=suchmaschine_url; }

```

Beispiel für altavista: suchmaschinen_url 'http://altavista.de/'
suchmaschinen_parameter 'cgi-bin/query?pg=q&what=web&q='

Hinweis: escape() setzt Umlaute und Leerzeichen in korrekte Darstellung um

.createTFOOT()	leeres TFOOT in einer Tabelle erzeugen und einbinden es darf nur 1 TFOOT zur Tabelle existieren siehe Objekt table
.createTHEAD()	leeres THEAD in einer Tabelle erzeugen und einbinden es darf nur 1 THEAD zur Tabelle existieren siehe Objekt table
decodeURI()	dekodiert einen Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde ersetzt die Methode unescape() welche deprecated ist ab IE 5.5
Beispiel:	alert(decodeURI("My%20phone%20 #%20is%20123-456-7890")); // erzeugt "My phone # is 123-456-7890"
decodeURIComponent()	dekodiert eine Komponente einer Uniform Ressource Identifier (URI), der mit der Methode encodeURI() erzeugt wurde ab IE 5.5
.Delete()	vorhandene Datei löschen (sonst Fehler erzeugt)
Beispiel:	var DateinameMitPfad = "c:\\test.txt"; var Dateisystem = new ActiveXObject("Scripting.FileSystemObject"); var Datei = Dateisystem.GetFile(DateinameMitPfad); Datei.Delete();
.Delete()	vorhandenen Ordner löschen (sonst Fehler erzeugt)
Beispiel:	var OrdnerName = "c:\\windows\\desktop\\"; var Dateisystem = new ActiveXObject("Scripting.FileSystemObject"); var Ordner = Dateisystem.GetFolder(OrdnerName); Ordner.Delete();
.deleteCaption()	CAPTION löschen aus Tabelle es darf nur 1 CAPTION zur Tabelle existieren siehe Objekt table
.deleteCell()	Zelle in die Zeile einer Tabelle löschen siehe Objekt table.tr



`.deleteData()` Teilkette aus einem Objekt entfernen

`.DeleteFile()` vorhandene Datei(en) löschen
Wenn Delete abbricht, so bleiben bisher erfolgte Löschaktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.DeleteFile("c:\\*.txt", true);
```

`.DeleteFolder()` vorhandene Ordner löschen, wobei es egal ist, ob Daten und/oder Unterordner im jeweiligen Ordner liegen oder nicht
Wenn Delete abbricht, so bleiben bisher erfolgte Löschaktionen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.DeleteFolder("c:\\*", true);
```

`.deleteRow()` Zeile löschen aus Tabelle
siehe Objekt table
siehe Objekt table.tBody
siehe Objekt table.tHead
siehe Objekt table.tFoot

`.deleteTFoot()` TFOOT der Tabelle löschen
es darf nur 1 TFOOT zur Tabelle existieren
siehe Objekt table
siehe Objekt table.tBody

`.deleteTHead()` THEAD der Tabelle löschen
es darf nur 1 THEAD zur Tabelle existieren
siehe Objekt table
siehe Objekt table.tBody

`.detachEvent()` Abschalten des Registrieren eines Events durch Eventhandler
wobei Registrierung mit Methode `.attachEvent()` aktiviert wurde
Abschalten = ordnet dem Window-Objekt das Event laut Parameter `event_bezeichner` zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)

Beispiel:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
function CursorNeu()
{
    if (event.srcElement == element)
    {
        normalColor      = style.color;
        runtimeStyle.color = "red";
        runtimeStyle.cursor = "hand";
    }
}

function CursorNormal()
{
    if (event.srcElement == element)
    {
        runtimeStyle.color = normalColor;
        runtimeStyle.cursor = "";
    }
}

function EventEntfernen()
{
    detachEvent ('onmouseover', CursorNeu);           // nicht () kodieren !!!
    detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
}

attachEvent ('onmouseover', CursorNeu);
attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

`.detachEvent()` Abschalten des Registrieren eines Events durch Eventhandler, wobei Registrierung mit Methode `.attachEvent()` aktiviert wurde
Achtung: Vor dem Neubelegen des Standard-Eventhandler diesen vor `.attachEvent()` retten und den Standard-Eventhandler **nach** `.detachEvent()` wieder einbinden (siehe Beispiel).



siehe Objekt window

Beispiel 1:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT=" EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor      = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
        window.onmouseover = Rette_Standard_Eventhandler_OnMouseOver;

        detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
        window.onmouseout = Rette_Standard_Eventhandler_OnMouseOut;
    }

    var Rette_Standard_Eventhandler_OnMouseOver = window.onmouseover;
    attachEvent ('onmouseover', CursorNeu);

    var Rette_Standard_Eventhandler_OnMouseOut = window.onmouseout;
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

Beispiel 2:

```
function ResizeHandler()                // Homepage neu laden
{window.history.go(0);}

// Standardhandler retten
var RetteHandler_Resize = window.onresize;

// und neu belegen
window.onresize = ResizeHandler;        // Homepage neu laden
                                         // ohne () kodieren, damit nicht sofort ausgeführt wird

// und Start des Abfangens vom resize-Event
window.attachEvent('onresize', ResizeHandler);

.....

// irgendwann abschalten der Eventüberwachung
window.detachEvent('onresize', ResizeHandler);

// und Standardhandler wieder einbinden
window.onresize = RetteHandler_Resize;
```

.doComponentRequest() Start des Download aller per .addComponentRequest() angeforderten Komponenten laut Puffer Behavior .style.clientCaps

.documentTimeToParentTime() Wert der Timeline des Dokumentes in den korrespondierenden Wert der Timeline der Eltern des Elementes konvertieren
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
```



```

</STYLE>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(ID_Excl.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onrepeat="innerText=parseInt(ID_Excl.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <t:EXCL ID="ID_Excl"
        CLASS="time_line_klasse"
        REPEATCOUNT="3"
    >
        <DIV ID="ID_Div"
            CLASS="time_line_klasse"
            BEGIN="2s"
            DUR="5s"
        >
        </DIV>
    </t:EXCL>
    <BR>
    <SPAN ID="ID_Span">Punkt auf der Timeline der Eltern:</SPAN>
    <BR>
    <BUTTON ID="ID_Button"
        CLASS="time_line_klasse"
        DUR="1"
        REPEATCOUNT="indefinite"
        onclick= "ID_Span.innerText=' Punkt auf der Timeline der Eltern: '
                + ID_Div.documentTimeToParentTime(
                    ID_Excl.currTimeState.activeTime
                );"
        onrepeat="innerText=' documentTimeToParentTime = '"
    >
        documentTimeToParentTime ermitteln
    </BUTTON>
</BODY>
</HTML>

```

.doImport() Start des Import des Programmcodes aus einer *.htc-Datei eines Behavior (Verhaltens) eines Elementes / Objektes
 der Import bewirkt die Erzeugung eines Elementes in der Collection namespaces, welche Behavior dynamisch verwaltet
 es kann binärer Code importiert werden
 Hinweis: Standard-Behavior des Internet Explorer besitzen keine *.htc-Datei und müssen nur importiert werden per #default#vordefinierter_name_des_behavior

Beispiel 1:

```

<HTML XMLNS:Mein_NamensRaum>
<HEAD>
    <SCRIPT LANGUAGE="JScript">
        document.namespaces("Mein_NamensRaum").doImport("#default");
    </SCRIPT>
</HEAD>

```

Beispiel 2:

```

<HEAD>
<HTML XMLNS: Mein_NamensRaum>
</HEAD>
<BODY onload=StartImport()>
<SCRIPT>
    var NamensRaumObjekt;

    function StartImport()

```



```

{
    // HTML XMLNS füllt die Collection namespaces
    //      da nur ein Namensraum erzeugt wurde, ist also nur 1 Element
    //      in der Collection namespaces

    // Namensraum laut Collection referenzieren
    NamensRaumObjekt = document.namespaces[0];

    // Import des Programmcodes aus der Datei test.htc starten (Download)
    NamensRaumObjekt.doImport("test.htc");

    // prüfen ob Import (Download) beendet ist
    if (NamensRaumObjekt.readyState != "complete")
    {
        // Import läuft noch also Endeereignis abfangen per Eventhandler
        //      wenn Import beendet ist, soll die Einbindung des
        //      Tags in das body-Objekt erfolgen
        NamensRaumObjekt.attachEvent("onreadystatechange",
                                     TagInDasBodyObjektEinbinden
                                     ); // Funktion ohne () kodieren !!!
    }
    else
    {
        // Import erledigt, also den Tag in das BODY-Objekt einbinden
        addTagNamesToBody();
    }

    return true;
}

function TagInDasBodyObjektEinbinden()
{
    // prüfen ob Import (Download) beendet ist
    if (NamensRaumObjekt.readyState != "complete")
    {return;}
    else
    {
        // Eventabfangen abschalten
        NamensRaumObjekt.detachEvent( "onreadystatechange",
                                       TagInDasBodyObjektEinbinden
                                       );

        // Tagobjekt erzeugen
        //      Tag ist ein Textelement
        var TagObjekt = document.createElement("Mein_NamensRaum: MeinTag");

        // Textelement füllen zum Rendern
        TagObjekt.innerText = "ElementBehavior";

        // und in das body-Objekt einbinden
        document.body.appendChild(TagObjekt);
    }
}
</SCRIPT>
</BODY>
</HTML>

```

Inhalt der test.htc

```

<public:component tagName=MeinTag>
<public:attach event=ondocumentready onevent=TagAktion()/>
</public:component>
<script>
    function TagAktion()
    {
        element.document.bgColor = "red";
    }
</script>

```

Beispiel 3: Import des Standard-Behavior .style.time2

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }

```




```
</STYLE>
</HEAD>
```

.doReadRequest() Lesezugriff auf alle vCard-Datenarten, die laut aktueller Request Queue vorgegeben sind
Lesezugriff auf einzelnen vCard-Wert per `.getAttribute()`
siehe Objekt `navigator.userProfile` und Autovervollständigung im Internet Explorer

Beispiel :

```
// Request Queue füllen

// es soll der Wert zu "vcard.displayname" ermittelt werden
navigator.userProfile.addReadRequest("vcard.displayname");
// es soll der Wert zu " vcard.gender " ermittelt werden
navigator.userProfile.addReadRequest("vcard.gender");

// Datenermitteln per Queue
navigator.userProfile.doReadRequest(12, "Daten von unbekannt verwendet");

// Queue leeren
navigator.userProfile.clearRequest();
```

.doScroll() Klick auf Scrollbar simulieren
Achtung: Scrollelemente müssen bereits vorhanden sein !

"scrollbarDown"	oder "down"	Down scroll Pfeil
"scrollbarHThumb"		horizontaler Scrollbalken
"scrollbarLeft"	oder "left"	Left scroll Pfeil
"scrollbarPageDown"	oder "pageDown"	Page-down Scrollbalken
"scrollbarPageLeft"	oder "pageLeft"	Page-left Scrollbalken
"scrollbarPageRight"	oder "pageRight"	Page-right Scrollbalken
"scrollbarPageUp"	oder "pageUp"	Page-up Scrollbalken
"scrollbarRight"	oder "right"	Right scroll Pfeil
"scrollbarUp"	oder "up"	Up scroll Pfeil
"scrollbarVThumb"		vertikaler Scrollbalken

Beispiel:

```
<HEAD>
<SCRIPT>
function ScrolleSeiteRechts()
{ document.body.doScroll("scrollbarPageRight");}

function ScrolleDown()
{ID_Textarea.doScroll("scrollbarDown");}

function ScrolleSeiteDown()
{ID_Textarea.doScroll("scrollbarPageDown");}
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick=" ScrolleSeiteRechts()">
    scrolle Seite rechts
</BUTTON>
<BR>
<BUTTON
    onclick=" ScrolleDown()"
    ondblclick=" ScrolleSeiteDown()"
>
    Texarea: Click = scrolle down Doppelklick = scrolle Seite down
</BUTTON>
<BR>
<BR>
<TEXTAREA ID="ID_Textarea" >
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
</TEXTAREA>
</BODY>
```

.dragDrop() prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
true Drag-Operation ist komplett erfolgreich durchgeführt
Hinweis zum Event `ondragstart`: immer true, wenn Maustaste losgelassen wurde
false Drag-Operation wurde nicht durchgeführt



<code>.DriveExists()</code>	prüfen auf Laufwerk im Dateisystem Laufwerk muss nicht bereits ein (z.B. Diskettenlaufwerk muss kein Medium enthalten)
Beispiel:	<pre>var DateiSystem = new ActiveXObject("Scripting.FileSystemObject"); alert(DateiSystem.DriveExists("A"));</pre>
<code>.duplicate()</code>	Zeiger auf eine Duplikat-Instanz eines Textbereiches liefern per <code>textrange</code> Objekt nur unter Windows 32-Bit Prüfung eines Textbereiches auf Kopie eines anderen Textbereiches per Methode <code>.inRange()</code> bzw. <code>.isEqual()</code>
<code>.elementFromPoint()</code>	liefert Referenz auf Objekt an Pixelpos relativ zum Fenster Fenster linke obere Ecke (0,0) Objekt muss Mausevents unterstützen
<code>.empty()</code>	Selektion löschen per <code>document.selection</code> Objekt des Dokumentes setzt zugleich Eigenschaft <div> <div><code>.item</code> auf null</div> <div>von <code>document.selection.textrange</code> Collection</div> </div> bzw. <div> <div><code>.type</code> auf "none"</div> <div>von <code>document.selection.controlRange</code> Collection</div> </div>
<code>encodeURIComponent()</code>	kodiert einen String als kompletten Uniform Ressource Identifier (URI): Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen <div> <div>, / ? : @ & = + \$ - _ . ! ~ * ' () #</div> <div>Buchstaben</div> <div>Ziffern</div> </div> ersetzt die Methode <code>escape()</code> welche deprecated ist ab IE 5.5 Beispiel: <pre>alert(encodeURIComponent("My phone # is 123-456-7890")); // erzeugt "My%20phone%20#%20is%20123-456-7890"</pre>
<code>encodeURIComponentComponent()</code>	kodiert einen Teil-String aus einem kompletten Uniform Ressource Identifier (URI): Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen <div> <div>, / ? : @ & = + \$ - _ . ! ~ * ' () #</div> <div>Buchstaben</div> <div>Ziffern</div> </div> ab IE 5.5
<code>.endElement()</code>	aktives Element auf der Timeline stoppen identisch mit Wirkung, wenn Endezeit des Elementes erreicht wird auf der Timeline bzw. die Dauer (Duration) des Elementes abgearbeitet wurde Alle Kinder des Elementes erhalten Information über Start des Elternelementes und sind somit korrekt auf der Timeline des Elternelementes. per Eigenschaft <code>.isActive</code> das Element auf Aktivsein prüfen siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code>
Beispiel:	<pre><HTML XMLNS:t="urn:schemas-microsoft-com:time"> <HEAD> <?IMPORT namespace="t" implementation="#default#time2"> <STYLE> .time_line_klasse { behavior: url(#default#time2) } </STYLE> </HEAD> <BODY> 0
 <t:EXCL ID="ID_Excl" REPEATDUR="indefinite"</pre>



```

>
    <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="3"
    >
        Erste Zeile
    </DIV>
    <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        BEGIN="3"
        DUR="3"
    >
        Zweite Zeile
    </DIV>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button" onclick="ID_Excl.endElement()">
    Stopp
</BUTTON>
</BODY>
</HTML>

```

`.endElementAt()` aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes
 ab Beginn der Timeline stoppen
 sinnvoll z.B. vor Erreichung des Ende der Dauer (Duration) des Elementes
 wenn das Elternelement diesen Zeitpunkt schon erreicht hat, so endet das Kindelement sofort,
 also ohne Zeitspanne
 per Eigenschaft `.isActive` das Element auf Aktivsein prüfen
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Stoppen()
    {
        // prüfen ob Elternobjekt, also body, den Endezeitpunkt schon erreicht hat
        // bzw. der Endewert falsch ist
        if ( (ID_Input.value <= document.body.currTimeState.activeTime)
            || (ID_Input.value==null)
        )
        {
            alert('Stoppwert falsch');
            ID_Input.value="";
            ID_Input.focus();
            return;
        }
        else
        { ID_Excl.endElementAt(ID_Input.value); }
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <SPAN ID="ID_Span2"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Excl.currTimeState.activeTime);"
    >

```



```

0
</SPAN>
<BR>
<t:EXCL ID="ID_Excl"
      BEGIN="0"
      CLASS="time_line_klasse"
      DUR="27"
>
  <DIV ID="ID_Div1"
      CLASS="time_line_klasse"
      BEGIN="0"
      DUR="5"
  >
    Erste Zeile
  </DIV>
  <DIV ID="ID_Div2"
      CLASS="time_line_klasse"
      BEGIN="5"
      DUR="5"
  >
    Zweite Zeile
  </DIV>
</t:EXCL>
<INPUT type="text" ID="ID_Input" SIZE="3" >
<BUTTON ID="ID_Button" onclick="Stoppen()">
  Start
</BUTTON>
</BODY>
</HTML>

```

escape()

kodiert einen String oder ein Literal in das Unicode-Format
 ab Javascript 1.5 (Netscape 6.x) deprecated und zu ersetzen durch die Methoden
 encodeURIComponent() und encodeURIComponent()
 Ersatz von Zeichen durch Escape-Sequenzen (UTF-8 Zeichen) der Form %xx
 Es werden alle Zeichen kodiert, also durch Escape-Sequenzen ersetzt, außer folgende Zeichen

```

. , / ? : @ & = + $ - _ . ! ~ * ' ( ) #
Buchstaben
Ziffern

```

URI (Uniform Resource Identifiers) werden nicht kodiert
 UTF-8 Zeichen: Teil des Unicode (0 bis 255)

Beispiel für Einbindung einer Suchmaschine:

```

var markierter_text=document.selection.createRange().text;
// oder var markierter_text=prompt('Suchbegriff: ');
var suchmaschinen_url='http:// .....';
var suchmaschinen_parameter='.....';

if (markierter_text)
{location.HREF=suchmaschinen_url + suchmaschinen_parameter + escape(markierter_text);}
else
{location.HREF=suchmaschine_url; }

```

Beispiel für altavista: suchmaschinen_url 'http://altavista.de/'
 suchmaschinen_parameter 'cgi-bin/query?pg=q&what=web&q='

eval()

parst einen String aus JavaScript-Anweisungen (kein HTML-Code) und führt diese sofort aus
 JavaScript-Anweisungen
 dürfen keine Referenz auf sich selbst haben (sonst endlose Rekursion !)
 können Referenzen auf existierende Objekte (Variablen) und deren Objektmethoden wie
 Objekteigenschaften enthalten z.B. für Wertzuweisung etc.
 Objekte (Variablen) erzeugen und instanzieren
 ab JavaScript 1.5 auch eval als Anweisung enthalten (Achtung: endlose Rekursionen
 vermeiden !)

Beispiele:

```

var Kette = eval("new String('2+2')"); // liefert Zeiger auf den String '2+2'
eval("var Kette = new String('2+2')"); // liefert nichts
// Kette hat den Wert '2+2'
eval("var Kette = new String('2+2'); alert(Kette);"); // liefert nichts, da alert nichts liefert
// Kette hat den Wert '2+2'

var StringLiteral = "2 + 2";

```



```
eval(StringLiteral)           // liefert Zeiger auf numerische Variable
                               // mit Wert 4
var Wert = eval("2+2");       // liefert Zeiger auf numerische Variable
                               // mit Wert 4
var StringObjekt = new String("2 + 2");
eval(StringObjekt)           // liefert Zeiger auf Kette "2 + 2"
```

Beispiel:

```
var w = 2;
var x = 39;
var y = "42";
var z = "42"
eval("w + x + 1");           // liefert Zeiger auf numerische Variable
                               // mit Wert 42
eval(y);                     // liefert Zeiger auf numerische Variable
                               // mit Wert 42
                               // liefert nicht Zeiger auf y, da der String
                               // "42" ausgewertet wird
eval(z);                     // liefert Zeiger auf String-Variable
                               // mit Wert '42'
                               // liefert nicht Zeiger auf z, da der String
                               // "42" ausgewertet wird
```

Beispiel:

```
var Kette = "var x=5;"
+ "if (x == 5)"
+ "{"
+ "alert('Meldung aus eval\nx ist 5');" // \n ist Zeilenumbruch-Zeichen
+ "x = 42;"
+ "}"
+ "else"
+ "{"
+ "x = 0;"
+ "};";

var Kette1 = "alert('Meldung aus document.write\nx ist ' + eval(Kette));"
document.write(Kette1);
```

Beispiel für Ersatz von eval() durch die Eigenschaft .innerHTML (nur IE):

```
<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
    ID_Div.innerHTML=Kette;           // wird sofort geparkt, also IMG-Tag ausgeführt
                                       // ( anstelle von eval()
                                       // bzw. document.write() )

    ID_IMG.src="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>
```

Beispiel:

```
if (VarUserAgent != "")
{
    for ( var i=0; i < 10; i++)
    {
        eval( 'if (VarUserAgent.indexOf("'" + i + ".") != -1)'
              + '{VarBrowser_Version_Haupt = ' + i + '};'
              );
    }
}
```



Beispiel:

```
function DatumHolen(Zeiger)
{
    var Kette = "Heute ist: ";
    Kette += Zeiger.getDate() + "/";
    Kette += (Zeiger.getMonth() + 1) + "/";
    Kette += Zeiger.getYear();

    return(Kette);
}

var Kette="Date";

eval(    "var Jetzt = new " + Kette + "();" // zur Laufzeit erzeugen
        "alert(DatumHolen(Jetzt));"      // Parameter ist zur Laufzeit bekannt
    );
```

.execCommand()

Kommando ausführen z.B. im aktuellen Dokument
 in aktueller Selektion
 im aktuellen Bereich
 erst nach dem kompletten Laden des Dokumentes zulässig
 Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
 Control = Element zur Steuerung analog zum HTML-Element (Tag)
 Input-Control = Element mit Eingabeeigenschaft

Beispiel 1:

```
<HTML>
<BODY>
    <H1 UNSELECTABLE="on">Demo</H1>
    <SCRIPT>
        function AddLink()
        {
            var SelektierterText = document.selection.createRange();

            if (!SelektierterText == "")
            {
                // Link erzeugen
                document.execCommand("CreateLink");

                if (SelektierterText.parentElement().tagName == "A")
                {
                    // markierten Text mit Eltern-Url ersetzen
                    SelektierterText.parentElement().innerText=
                        SelektierterText.parentElement().href;

                    // Vordergrundfarbe setzen im Dokument
                    document.execCommand(
                        "ForeColor", "false", "#FF0033");
                }
            }
            else
            { alert("Bitte im blauen Text selektieren !"); }
        }
    </SCRIPT>
    <P UNSELECTABLE="on">
        Selektiere (markiere) im nachfolgenden blauen Text die Stelle mit
        dem Text MARKIERE_MICH.<BR>
        Danach auf das Button klicken.<BR>
        Anstelle von MARKIERE_MICH im blauen Text erscheint dort
        nun eine Url.
    </P>
    <P STYLE="color=#3366CC">
        Meine beliebteste Webseite MARKIERE_MICH bitte besuchen !
    </P>
    <BUTTON onclick="AddLink()" UNSELECTABLE="on">Klick</BUTTON>
</BODY>
</HTML>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
    function HandlerFuerOnMoveStart()
    {
```



```

// anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
var ZeigerAufObjektMitEvent = event.srcElement;

ZeigerAufObjektMitEvent.style.backgroundColor = "green";
ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandlerFuerOnMoveEnd()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

// 2-D Positionierung einschalten
document.execCommand("2D-position",false,true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offserTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE=
            "position:absolute;width:300px;height:100px; background-color:red;"
        >
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

.execScript() Pendant zur Methode eval()
 Script wird sofort ausgeführt
 Script in diversen Sprachen möglich
 alle vorhandenen-globalen Variablen ansprechbar
 siehe Objekt window

Beispiel.: window.execScript('alert("Hallo");', 'JScript')

.Exists() prüfen auf Vorhandensein eines Datenschlüssels im Dictionary

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Meldung ob Date vorhanden ist
    alert(SchlüsselVorhanden(DatenSchluessel));
}

```

.exp() liefert den Wert von E hoch zahl
 siehe Script-Objekt Math



<code>.expand()</code>	Plain-Text als Teil eines Textbereiches derart ausdehnen, dass er komplett die Dimension des Elternobjektes (Container-Objektes) einnimmt per <code>textrange</code> Objekt nur unter Windows 32-Bit
<code>.expression()</code>	Wert einer Style-Eigenschaft per STYLE-Attribut-Wert in HTML als Ausdruck definieren für spätere Berechnung per Methode <code>getExpression()</code> Ausdruck nur als Script kodierbar DOM wird geändert siehe auch Methode <code>.setExpression()</code>

In nachfolgenden Beispielen wurde aus Platzgründen die Zeichenkette von STYLE umgebrochen, was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

Beispiel 1:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE = "JScript">
    function width_init()
    {
        ID_Span.style.setExpression( "width",
                                     " trueBlueSpan.style.pixelWidth
                                     + oldYellowSpan.style.pixelWidth
                                     ",
                                     "jscript"
                                   );
    }

    function Berechne()
    {alert(ID_Span.style.getExpression("width");}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
          STYLE="background-color:lightgreen;
                width:expression( trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                )
                "
    >
    </SPAN>
    <BUTTON onclick= Berechne();>
</BODY>
</HTML>
```

Beispiel 2:

```
ID_Span.style.setExpression("height", "document.style.fontSize + 13");
ID_Span.style.setExpression("width", "document.body.style.fontSize");
<SPAN ID="ID_Span"
      STYLE="background-color:lightgreen;
            width:expression( trueBlueSpan.style.pixelWidth
                            + oldYellowSpan.style.pixelWidth
                            )
            "
>
</SPAN>
```

`.FileExists()` prüfen auf Datei

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.FileExists("c:\\test.txt"));
```

`.findText()` Text im gesamten Textbereich suchen
per `textrange` Objekt
nur unter Windows 32-Bit
für Nutzung einer Textmarke siehe Methoden `.getBookmark()` und `.moveToBookmark()`

`.fire()` ein in der HTC-Komponente definiertes Event erzeugen und an das HTML-Dokument weiterreichen
siehe Objekt `element` und HTC-Datei

Beispiel:

test.htc enthält:

```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>
```



```

<SCRIPT LANGUAGE="JScript">
    function Berechne()// Bezug über ID_Event
    {
        var EventObjekt    = createEventObject();
        EventObjekt.result = "Wert";

        ID_Event.fire (EventObjekt);
    }
</SCRIPT>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
    @media all { privater_tag_namensraum\;privater_tag {behavior:url(test.htc)} }
</STYLE>
</HEAD>
<BODY>
<privater_tag_namensraum:privater_tag
    ID="ID_privater_tag"
    onResultChange="ID_Div.innerText=window.event.result"
>
    <TABLE>
        <TR>
            <DIV
                ID="ID_Div"
                ALIGN=RIGHT
                STYLE="border: '.025cm solid gray'"
            >
                0.
            </DIV>
        </TR>
        <TR>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 7 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 8 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 9 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" / ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" C ">
            </TD>
        </TR>
        <TR>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 4 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 5 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 6 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" * ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" % " DISABLED>
            </TD>
        </TR>
        <TR>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 1 ">
            </TD>
            <TD>
                <INPUT TYPE=BUTTON VALUE=" 2 ">
            </TD>

```



```

        <TD>
        <INPUT TYPE=BUTTON VALUE=" 3 ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" - ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
    </TD>
</TR>
<TR>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" 0 ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" +/- ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" . ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" + ">
    </TD>
    <TD>
        <INPUT TYPE=BUTTON VALUE=" = ">
    </TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

`.fireChange()` erzeugt Event `onpropertychange` und gibt es an das HTML-Dokument, sobald der Wert der Eigenschaft per Aufruf der Funktion laut Attribut `PUT` verändert wird
 Aufruf von `.fireChange()` muss programmiert werden
 siehe Objekt `element` und HTC-Datei

Beispiel:

```

<PUBLIC:PROPERTY
    ID="ID_HTC_Komponente"
    NAME="HTC_Komponente"
    PUT="Eigenschaft_Schreiben"
    GET="Eigenschaft_Lesen"
/>

<SCRIPT LANGUAGE="JScript">
    var Eigenschaft_Wert =null;

    function Eigenschaft_Schreiben (Wert)
    {
        // Wert zuweisen
        Eigenschaft_Wert = Wert;

        // und Event erzeugen
        ID_HTC_Komponente.fireChange();
    }

    function Eigenschaft_Lesen ()
    {return Eigenschaft_Wert; }
</SCRIPT>

```

`.fireEvent()` ein Event auslösen und zugehöriges Objekt **object**.event erzeugen, wobei **object** das Event auslöst
 mit Aufruf der Methode werden folgende Eigenschaften des even Objektes automatisch belegt:

object .event.cancelBubble	auf false
object .event.returnValue	auf true
object .event.srcElement	mit Zeiger des eventerzeugenden Objektes belegt
object .event.type	mit dem Eventtyp belegt

`object` kann entfallen wenn aktuelles Fenster gemeint ist, also `window`
 zeiger_auf_fenster

siehe `.createEventObject()`

Beispiel 1:

```

<HTML>
<HEAD>
<SCRIPT>

```



```

function EventAusloesen()
{
    ID_Div.innerText = "onmouseover-Event wurde ausgelöst!";

    // Event onclick auslösen
    ID_Button.fireEvent("onclick");
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div" onmouseover=" EventAusloesen();">
    Bitte hier Mouse over!
</DIV>
<BUTTON ID="ID_Button" ONCLICK="this.innerText='onclick-Event wurde ausgelöst!' ">
    Klick mich nicht , denn ich werde durch den DIV geklickt!
</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<SCRIPT LANGUAGE="JScript">
function DurchReichen()
{
    if (window.event.shiftKey)
    {window.event.cancelBubble = true;}
}

function Anzeigen()
{
    if (window.event.srcElement.tagName == "IMG")
    {alert(window.event.srcElement.src);}
}
</SCRIPT>
<BODY onclick="Anzeigen()">
<IMG SRC="test.gif" onclick="DurchReichen()">

```

.firstPage() erste Seite des Datasets in Tabelle anzeigen
 Eigenschaft .dataPageSize muss belegt worden sein
 siehe Objekt table

Beispiel:

Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815
	Willmann;Theo;1234
	Xantippe;Isa;5678
	Arktin;Richard;1055

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) {Kette = "+";}

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
    }

```



```

        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            {ID_Datenbank.recordset.MovePrevious();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition ==1)
        {alert("Erster Datensatz erreicht!");}
    }

    // -->
    </SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
>
    <THEAD>
        <TR>
            <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
            <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
            <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD><DIV DATAFLD="vorname"></DIV></TD>
            <TD><DIV DATAFLD="name"></DIV></TD>
            <TD><DIV DATAFLD="telefon"></DIV></TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->
</BODY>
</HTML>

```

.fixed()

HTML-Tag <TT> erzeugen in der Form <TT>text</TT>



Beispiel: siehe Script-Objekt String

```
var StringLiteral    ="Text der TT wird"
var HTML_Kette      = StringLiteral.fixed();
HTML_Kette          = "Text der TT wird".fixed();
```

entspricht <TT>Text der TT wird</TT>

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.floor() nächste ganze Zahl unterhalb zahl ermitteln
Bsp: floor(1.1) ergibt 1
floor(1) ergibt 1
siehe Script-Objekt Math

.focus() Focus setzen und Focus-Event auslösen
nur nach dem kompletten Laden des Dokumentes
vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

Beispiel: <BODY onload="document.body.focus();>

Beispiele für Abschaltung der automatischen Umrandung von angeklickten Objekten durch Fokussierung des BODY:

<BODY onclick="if (document.all){body.focus();}"> </BODY>

.focus() Focus setzen und Event onfocus auslösen
nur nach dem kompletten Laden des Dokumentes
vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
siehe Objekt window

.FolderExists() prüfen auf Ordner

Beispiel: var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.FolderExists("c:\\test\\"));

.fontcolor() HTML-FONT als HTML-Kette erzeugen in der Form "text"
und ohne weitere Attribute ID etc.
siehe Script-Objekt String

Beispiel: var StringLiteral ="Sichtbarer Text"
var HTML_Kette = StringLiteral.fontcolor("WertDesCOLORAttributes");
HTML_Kette = "Sichtbarer Text".fontcolor("WertDesCOLORAttributes");

entspricht Sichtbarer Text

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.fontsize() HTML-FONT als HTML-Kette erzeugen in der Form "text"
und ohne weitere Attribute ID etc.
siehe Script-Objekt String

Beispiel: var StringLiteral ="Sichtbarer Text"
var HTML_Kette = StringLiteral.fontsize("WertDesSIZEAttributes");
HTML_Kette = "Sichtbarer Text".fontsize("WertDesSIZEAttributes");

entspricht Sichtbarer Text

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.forward() Aktivierung einer zur aktuell besuchten Seite im Verlauf des Surfens danachliegenden Seite
leider ohne Sprünge
entspricht Druck des Forward-Buttons

.fromCharCode() String-Literal erzeugen aus Folge von Unicoden
Unicode von 0 bis 65535, wobei
0 bis 127 der ASCII-Zeichensatz ist
0 bis 255 der ISO-Latin-1-Zeichensatz ist
siehe Script-Objekt String

Beispiel 1:




```

        <OPTION>woher ist afterBegin
        <OPTION>woher ist beforeEnd
        <OPTION>woher ist afterEnd
    </SELECT>
    <INPUT TYPE="button" VALUE="Anzeigen" onclick=" Anzeigen()">

```

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
 DOM nicht geändert

Beispiel:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachenames definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT    ID="ID_Input"
               CLASS="user_data_speicher_klasse"
               TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten"    onclick="InputLaden()">
    </FORM>

```



```
</BODY>
</HTML>
```

`.getAttribute()` einzelnen vCard-Wert lesen ohne Request Queue
siehe Objekt `navigator.userProfile` und Autovervollständigung im Internet Explorer

Beispiel:

```
// lesen vom Wert zu "vcard.displayname"
var Name = navigator.userProfile.getAttribute("vcard.displayname");

// lesen vom Wert zu "vcard.gender"
var Gender = navigator.userProfile.getAttribute("vcard.gender");
```

`.getAttributeName()` Name des mit dem MediaItem Objekt verbundenen Attributes liefern
z.B. zur Feststellung, welche Attribute eine Advanced Stream Redirector (ASX)-Datei hat
wie `abstract` oder `author` oder `copyright`

Beispiel für Aufbau eines Eintrages in einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
<entry>
  <title>Testitel</title>
  <author>Testautor</author>
  <copyright>Test 2002</copyright>
  <abstract>WAV Datei</abstract>
  <ref href=""></ref>
  <banner href = "Testbild.gif" >
    <moreinfo href = "Test.doc"></moreinfo>
    <abstract>besuche www.test.de</abstract>
  </banner>
</entry>
</ASX>
```

Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

```
.getItemInfo()
.getAttributeName()
```

Eigenschaften nutzbar:

```
.attributeCount
```

Es ist also vorher die Eigenschaft `.playState` auf Werte > 10 zu prüfen.
siehe Behavior `.style.mediaBar`

`.getAttributeNode()` Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf `attribute.name` Eigenschaft.
Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht
Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt
Wert des Attributes wird somit über die Referenz laut DOM erreichbar
DOM nicht geändert

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
  function ToolTipKnotenErmitteln()
  {
    return (ID_Div.getAttributeNode("TITLE"));
  }
</SCRIPT>
</HEAD>
<BODY onload="ToolTipKnotenErmitteln();" >
  <DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>
```

`.GetBaseName()` absoluten Pfad einer Datei im Pfad liefern
Datei muss nicht existieren

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetBaseName("../test\\text.txt"));
```

`.getBookmark()` Textmarke (Bookmark) im Textbereich setzen
Textmarke kann mit Methode `.moveToBookmark()` anpositioniert werden
per `textrange` Objekt
nur unter Windows 32-Bit

`.getBoundingClientRect()` Referenz auf `TextRectangle`-Objekt im Element holen



Beispiel:

```

<HEAD>
<SCRIPT>
    var ZeigerAufTextRectangleCollection;
    var Index =0;

    function Anzeigen(Zeiger) // Zeiger auf den einzigsten DIV mit Text also mit Textbereich
    {
        // aktuelle Collection der Rectangle von Div0 referenzieren:
        // Das ist nötig nach resize des Fensters, da resize leider nicht automatisch erkannt wird !
        ZeigerAufTextRectangleCollection = Zeiger.getClientRects();

        // Anzahl der Elemente ermitteln, also Anzahl der im Browser dargestellte Zeilen,
        // wobei es egal ist, ob diese per <BR> erzeugt wurden oder nicht
        // Hinweis: Zeilenlänge hängt auch von der Fensterbreite ab
        // (automatischer Umbruch)
        // Jede Zeile besitzt ihr eigenes Rectangle
        AnzahlTextRectangle = ZeigerAufTextRectangleCollection.length;

        // aktuellen Index prüfen ob letzte Zeile bereits erreicht wurde
        if (Index > AnzahlTextRectangle -1) // Index ab 1, Anzahl ab 1
        {
            // es wurde die letzte Zeile erreicht

            // Div2 unsichtbar machen also entfärben
            // Hinweis: Div2 liegt auf dem Rectangle von Div0
            ID_Div2.style.display="none";

            // rücksetzen des Index, also mit erster Zeile weitermachen
            Index = 0;
        }

        // Rechteck der aktuellen Zeile ermitteln unter Beachtung eines eventuellen Scrollens
        var PosRechts = ZeigerAufTextRectangleCollection[Index].right + ID_Body.scrollLeft;
        var PosLinks = ZeigerAufTextRectangleCollection[Index].left + ID_Body.scrollLeft;
        var PosOben1 = ZeigerAufTextRectangleCollection[Index].top + ID_Body.scrollTop;

        // und Div1 auf die Zeile positionieren, also Zeile einfärben
        ID_Div1.style.top = PosOben1;
        ID_Div1.style.width = (PosRechts - PosLinks) - 5;
        ID_Div1.style.display = 'inline';

        // aktuelle Position des Rechteckes von DIV0 ermitteln unter Beachtung eines eventuellen
        // Scrollens
        PosRechts = Zeiger.getBoundingClientRect().right + ID_Body.scrollLeft;

        PosLinks = Zeiger.getBoundingClientRect().left + ID_Body.scrollLeft;
        var PosOben2 = Zeiger.getBoundingClientRect().top + ID_Body.scrollTop;

        // und Div2 überlagern positionieren
        ID_Div2.style.top = PosOben2;
        ID_Div2.style.width = (PosRechts - PosLinks) - 5;
        ID_Div2.style.height = PosOben1 - PosOben2;

        // aber nur einfärben, wenn mindestens 1 Zeile bereits eingefärbt wurde
        if (Index > 0){ ID_Div2.style.display = 'inline';}

        // Rectangle der nächsten Zeile einstellen
        Index++;
    }
</SCRIPT>
</HEAD>
<BODY ID="ID_Body">
    <DIV ID="ID_Div0"
        onclick=" Anzeigen(this)"
    >
        klicke
        <BR>
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
    </DIV>

```



```

ABCDEF GHIJ KLMNOPQRST UVWXYZ
1234567890123456789012345678901234567890
</DIV>
<DIV ID="ID_Div1"
STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:yellow; display:none"
>
</DIV>
<DIV ID="ID_Div2"
STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:beige; display:none"
>
</DIV>
</BODY>

```

.getClientRects() Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster
 Feld mit Index als Integer ab 0
 pro Eintrag ein Rectangle

Beispiel:

```

<HEAD>
<SCRIPT>
var ZeigerAufTextRectangleCollection;
var Index =0;

function Anzeigen(Zeiger) // Zeiger auf den einzigsten DIV mit Text also mit Textbereich
{
    // aktuelle Collection der Rectangle von Div0 referenzieren:
    // Das ist nötig nach resize des Fensters, da resize leider nicht automatisch erkannt wird !
    ZeigerAufTextRectangleCollection = Zeiger.getClientRects();

    // Anzahl der Elemente ermitteln, also Anzahl der im Browser dargestellte Zeilen,
    // wobei es egal ist, ob diese per <BR> erzeugt wurden oder nicht
    // Hinweis: Zeilenlänge hängt auch von der Fensterbreite ab
    // (automatischer Umbruch)
    // Jede Zeile besitzt ihr eigenes Rectangle
    AnzahlTextRectangle = ZeigerAufTextRectangleCollection.length;

    // aktuellen Index prüfen ob letzte Zeile bereits erreicht wurde
    if (Index > AnzahlTextRectangle -1) // Index ab 1, Anzahl ab 1
    {
        // es wurde die letzte Zeile erreicht

        // Div2 unsichtbar machen also entfärben
        // Hinweis: Div2 liegt auf dem Rectangle von Div0
        ID_Div2.style.display="none";

        // rücksetzen des Index, also mit erster Zeile weitermachen
        Index = 0;
    }

    // Rechteck der aktuellen Zeile ermitteln unter Beachtung eines eventuellen Scrollens
    var PosRechts = ZeigerAufTextRectangleCollection[Index].right + ID_Body.scrollLeft;
    var PosLinks = ZeigerAufTextRectangleCollection[Index].left + ID_Body.scrollLeft;
    var PosOben1 = ZeigerAufTextRectangleCollection[Index].top + ID_Body.scrollTop;

    // und Div1 auf die Zeile positionieren, also Zeile einfärben
    ID_Div1.style.top = PosOben1;
    ID_Div1.style.width = (PosRechts - PosLinks) - 5;
    ID_Div1.style.display = 'inline';

    // aktuelle Position des Rechteckes von DIV0 ermitteln unter Beachtung eines eventuellen
    // Scrollens
    PosRechts = Zeiger.getBoundingClientRect().right +
ID_Body.scrollLeft;

    PosLinks = Zeiger.getBoundingClientRect().left + ID_Body.scrollLeft;
    var PosOben2 = Zeiger.getBoundingClientRect().top + ID_Body.scrollTop;

    // und Div2 überlagern positionieren
    ID_Div2.style.top = PosOben2;
    ID_Div2.style.width = (PosRechts - PosLinks) - 5;
    ID_Div2.style.height = PosOben1 - PosOben2;

    // aber nur einfärben, wenn mindestens 1 Zeile bereits eingefärbt wurde
    if (Index > 0){ ID_Div2.style.display = 'inline';}

    // Rectangle der nächsten Zeile einstellen

```



```

        Index++;
    }
</SCRIPT>
</HEAD>
<BODY ID="ID_Body">
    <DIV ID="ID_Div0"
        onclick=" Anzeigen(this)"
    >
        klicke
        <BR>
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890123456789012345678901234567890
        ABCDEFGHIJKLMNOPQRSTUVWXYZ
        123456789012345678901234567890
    </DIV>
    <DIV ID="ID_Div1"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:yellow; display:none"
    >
    </DIV>
    <DIV ID="ID_Div2"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:beige; display:none"
    >
    </DIV>
</BODY>

```

.getComponentVersion() Version einer Komponente
 Behavior .style.clientCaps

.getData() Clipboard auslesen
 Anwendung für Ereignisse oncopy oder oncut
 Handler muss liefern window.event.returnValue=false;
 "Text" Daten im Text-Format auslesen
 "URL" Daten im Url-Format auslesen
 "File" Daten im File-Format auslesen
 "HTML" Daten im HTML-Format auslesen
 "Image" Daten im Image-Format auslesen: Es wird Pfad geliefert

.getDate() Tag des Monats in lokaler Zeit liefern
 Script-Objekt Date
 Integer 1 bis 31

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                        + "="
                        + escape(Value)
                        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                        + "="
                        + escape(Value)
                        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //        erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie

```



```

//      Aufbau   Cookie_Name = Cookie_Wert
//                      Separator ist Gleichheitszeichen
//      Index 0 für Cookie_Name
//      Index 1 für Cookie_Wert
var CookieNameUndWertAlsFeld;

// CookieFeld auslesen und auf Cookie laut Name prüfen
var Gefunden=false;
var Index = 0;
do
{
    // aktuelles Cookie lesen
    CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

    // und auf Name prüfen
    if (Name == CookieNameUndWertAlsFeld [0])
    {
        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while (      ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachename festlegen
//      es können diverse Cachennamen definiert und somit Versionen von Cache
//      verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
//      es können diverse Attribute definiert und somit Versionen von Input-Daten
//      verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++++ Zeitstempel ++++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++++ Daten chachen ++++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;

    // Attribut instanzieren und mit Daten füllen
    InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

    // und Cache saveen

```



```

        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getDay()
    Tag der Woche in lokaler Zeit liefern
    Script-Objekt Date
    Integer 0 bis 6
    0 ist Sonntag
    6 ist Samstag

```

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)    // Name und Wert sind Strings
    {
        var document.cookie =    Name
                                + "="

```




```

+ escape(Value)
+ "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while ( ( !Gefunden )
        && ( Index < AnzahlCookies )
    );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachename festlegen
// es können diverse Cachennamen definiert und somit Versionen von Cache
// verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
// es können diverse Attribute definiert und somit Versionen von Input-Daten
// verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

```



```

// und als UTC-Format erzeugen
var ZeitstempelUTC = Zeitstempel.toUTCString();

// und Zeitstempel dem Input-Objekt verpassen
ID_Input.expires = ZeitstempelUTC;

// ++++++ Daten chachen ++++++
// aktuelle Daten holen laut Input-Objekt
var InputDaten = InputDatenObjekt.value;

// Attribut instanzieren und mit Daten füllen
InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

// und Cache saveen
InputDatenObjekt.save(FreierCacheName);
}

function InputLaden()
{
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
}

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
            + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
            );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

.GetDrive() Objekt FileSystemObject.Drive anhand des Laufwerksbuchstaben eines existierenden Laufwerkes erzeugen (auch bei Netzlaufwerk) sonst wird Fehler erzeugt

Beispiel:

```

var DateiSystem                      = new ActiveXObject("Scripting.FileSystemObject");
var Laufwerk                         = DateiSystem.GetDrive("C");

```



```

var Laufwerk_VolumeName = Laufwerk.VolumeName;
var Laufwerk_TotalerPlatz = Laufwerk.TotalSize;
alert(Laufwerk + " " + Laufwerk_VolumeName + " " + Laufwerk_TotalerPlatz);

```

`.GetDriveName()` Laufwerksbuchstabe aus Pfad ermitteln in der Form "x:" mit x für Laufwerk
 Pfad muss nicht existieren
 muss Laufwerksbuchstaben enthalten
 wird nicht auf Syntax geprüft

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetDriveName("c:\\.\\.*\\test\\t***");

```

`.getElementById()` Referenz auf das im Dokument ZUERST gefundene Objekt laut ID (analog zum ID-Attribut) liefern
 Achtung: Objekte, die kein ID besitzen, werden nicht erfasst !
 Für Verwaltung per NAME (analog zum NAME-Attribut):
 siehe Methode `getElementsByName()`
 Für Verwaltung per Tag-Name
 siehe Methode `.getElementsByTagName()`
 wenn mehrere Elemente mit ein und demselben ID, so das ERSTE Element von diesen referenziert
Eine Referenzierung einer Collection der Objekte mit gemeinsamen ID ist leider nicht möglich. Deswegen der strenge Hinweis:
 In der Regel werden ID vom Programmierer objektweise getrennt vergeben, es sei denn, man will bewusst eine Gruppe von Objekten (z.B. `RadioBox`) verwaltbar machen und kennt diese Objekte. Die maschinelle Analyse eines fremden Dokumentes mit der Methode `.getElementById()` ist nicht möglich.
 DOM nicht geändert
`getElementById()` ist der Standard, um Referenz zu ermitteln, und den alle Browser können müssen
 ID-Attribut ermöglicht zwar den bequemen Ersatz, aber nur, wenn das ID-Attribut überhaupt im Objekt implementiert ist.
 Wird mit `createElement()` gearbeitet, so ist der Zeiger auf Das Objekt global speicherbar (in einem Feld), so dass `getElementByXXXX` nicht mehr benötigt wird.
 Alternativ sind Collectionen von `document` etc. nutzbar.
 Besonders faule Programmierer nutzen intensiv `getElementByXXXXXX` und blähen den Quellcode auf (Literaten als Programmierer)

Beispiel:

```

<SCRIPT>
function Referenziere()
{
    var Zeiger=document.getElementById("ID_Div");
}
</SCRIPT>
<DIV ID="ID_Div">Test</DIV>
<INPUT TYPE="button" VALUE=" Referenziere" onclick=" Referenziere()">

```

`.getElementsByName()` Referenz auf ein Feld (Collection) aller im Dokument befindlichen Objekte mit gemeinsamen NAME (analog zum Attribut NAME) liefern
 Achtung: Objekte, die kein NAME besitzen, werden nicht erfasst !
 Für Verwaltung per ID (analog zum ID-Attribut):
 siehe Methode `getElementById()`
 Für Verwaltung per Tag-Name
 siehe Methode `.getElementsByTagName()`
 DOM nicht geändert
 ID-Attribut ermöglicht zwar den bequemen Ersatz, aber nur, wenn das ID-Attribut überhaupt im Objekt implementiert ist.
 Wird mit `createElement()` gearbeitet, so ist der Zeiger auf Das Objekt global speicherbar (in einem Feld), so dass `getElementByXXXX` nicht mehr benötigt wird.
 Alternativ sind Collectionen von `document` etc. nutzbar.
 Besonders faule Programmierer nutzen intensiv `getElementByXXXXXX` und blähen den Quellcode auf (Literaten als Programmierer)

Beispiel:

```

<SCRIPT>
function Referenziere()
{
    var FeldDerInput=document.getElementsByName("GemeinsamerName");
}
</SCRIPT>
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="text" NAME="GemeinsamerName">
<INPUT TYPE="button" NAME="Button" VALUE=" Referenziere" onclick=" Referenziere()">

```

`.getElementsByTagName()` Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.
 Hinweis: Natürlich kann auch das `document`-Objekt so verarbeitet werden (beachte dabei `document.all` Collection)



Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !

Für Verwaltung per ID (analog zum ID-Attribut):

siehe Methode `getElementsById()`

Für Verwaltung per NAME (analog zum NAME-Attribut):

siehe Methode `getElementsByName()`

DOM nicht geändert

`getElementByTagName()` ist der Standard, um Referenz zu ermitteln, und den alle Browser können müssen ID-Attribut ermöglicht zwar den bequemeren Ersatz, aber nur, wenn das

ID-Attribut überhaupt im Objekt implementiert ist.

Wird mit `createElement()` gearbeitet, so ist der Zeiger auf Das Objekt global

speicherbar (in einem Feld), so dass `getElementByXXXX` nicht mehr benötigt wird.

Alternativ sind Collectionen von `document` etc. nutzbar.

Besonders faule Programmierer nutzen intensiv `getElementByXXXXXX` und blähen den Quellcode auf (Literaten als Programmierer)

Beispiel 1:

```
var DIV_KinderZeigerFeld = document.body.getElementsByTagName("DIV");
```

Hinweis: entspricht `var DIV_KinderZeigerFeld = document.body.all.tags("DIV");`

Beispiel 2:

```
<SCRIPT>
var Feld_Span = ID_DivEltern.getElementsByTagName("SPAN");
// alle SPAN-Kinder referenzieren
</SCRIPT>
<DIV ID="ID_DivEltern">
  <SPAN>
    Span-Kind von ID_DivEltern
  <DIV>
    Div-Kind vonDivEltern-Span
    <SPAN>
      Span-Kind vonDivEltern-Span-Div
    </SPAN>
  </DIV>
  </SPAN>
</DIV>
```

Beispiel 3:

```
<SCRIPT>
function Anzeige()
{
  var ZeigerAufOnClickEventQuelle=event.srcElement;
  var Feld =
    ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
  alert(
    "Anzahl LI : "
    + Feld.length
    + "\nErster Eintrag: "
    + Feld [0].childNodes[0].nodeValue
  );
}
</SCRIPT>
<UL onclick="Anzeige()">
  <LI>Menuepunkt 1
  <UL>
    <LI> Menuepunkt 1.1
    <OL>
      <LI> Menuepunkt 1 1.1
      <LI> Menuepunkt 1 1.2
    </OL>
    <LI> Menuepunkt 1.2
    <LI> Menuepunkt 1.3
  </UL>
  <LI> Menuepunkt 2
  <UL>
    <LI> Menuepunkt 2.1
    <LI> Menuepunkt 2.3
  </UL>
  <LI> Menuepunkt 3
</UL>
```

`.getExpression()`

Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern

Style-Eigenschaft ist per Methoden

`expression()` oder `setExpression()`

zu definieren

DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout

(nach dem eventuellen expliziten Dokument-Refresh)



In nachfolgenden Beispielen wurde aus Platzgründen die Zeichenkette von STYLE umgebrochen, was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

Beispiel 1:

```
<SPAN ID="ID_Span"
      STYLE="background-color:lightgreen;
            width:expression( trueBlueSpan.style.pixelWidth
                              + oldYellowSpan.style.pixelWidth
                              )
      ">
</SPAN>
<BR>
<BUTTON onclick=alert(ID_Span.style.getExpression("width"));>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE = "JScript">
    function width_init()
    {
        ID_Span.style.setExpression("width", "
                                     trueBlueSpan.style.pixelWidth
                                     + oldYellowSpan.style.pixelWidth",
                                     "jscript"
                                     );
    }

    function Berechne()
    {alert(ID_Span.style.getExpression("width");}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
          STYLE="background-color:lightgreen;
                width:expression( trueBlueSpan.style.pixelWidth
                                  + oldYellowSpan.style.pixelWidth
                                  )
          ">
    </SPAN>
    <BUTTON onclick= Berechne();>
</BODY>
</HTML>
```

.GetExtensionName() Suffix einer Datei liefern ohne Trenner "."
Datei muss nicht existieren

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetExtensionName("../test\\text.txt"));
```

.GetFile() Objekt FileSystemObject.File anhand Dateinamen einer existierenden Datei erzeugen
(sonst wird Fehler erzeugt)

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile("../test\\text.txt");
alert(Datei);
```

.GetFileName() Name einer Datei mit Suffix und mit Trenner "."
Datei muss nicht existieren

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetFileName("../test\\text.txt"));
```

.GetFolder() Objekt FileSystemObject.Folder anhand Ordernamen eines existierenden Ordners erzeugen
sonst wird Fehler erzeugt

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Ordner = DateiSystem.GetFolder("../test\\");
alert(Ordner);
```

.getFullYear() Jahreszahl vierstellig in lokaler Zeit liefern
Script-Objekt Date



Integer, maximal **1970 + bzw. -285,616 Jahre**

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für Cookie_Name
    //      Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies)
            );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachennamen definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

```



```

// Cache-Attribut festlegen
//      es können diverse Attribute definiert und somit Versionen von Input-Daten
//      verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitpunktUTC = Zeitpunkt.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitpunktUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;

    // Attribut instanzieren und mit Daten füllen
    InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

    // und Cache saveen
    InputDatenObjekt.save(FreierCacheName);
}

function InputLaden()
{
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
}

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;
    }

```




```

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " + parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getHours()           Stunden in lokaler Zeit liefern
                       Script-Objekt Date
                       Integer 0 bis 23

```

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für Cookie_Name
    //      Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (      ( !Gefunden )
              && ( Index < AnzahlCookies )
              );

    return CookieWert;
}
</SCRIPT>

```



Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachennamen definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitpunktUTC = Zeitpunkt.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitpunktUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

```



```

function fnInit()
{
    var AnzahlMillisekundenProTag = 86400000;

    var DatumDokumentErzeugung = new Date(document.fileCreatedDate);

    var DatumHeute = new Date();

    var TagesDifferenz = ( DatumHeute.getTime()
                        - DatumDokumentErzeugung.getTime()
                        )
                        / AnzahlMillisekundenProTag;

    alert( "Dokument erzeugt am " + DatumDokumentErzeugung
          + "\n..... also vor " + parseInt(TagesDifferenz) + " Tagen"
          );
}
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

`.getItemInfo()` Wert des mit dem MediaItem Objekt verbundenen Attributes liefern
Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:

- `.getItemInfo()`
- `.getAttributeName()`

Eigenschaften nutzbar:

- `.attributeCount`

Es ist also vorher die Eigenschaft `.playState` auf Werte > 10 zu prüfen.
siehe Behavior `.style.mediaBar`

`.getMilliseconds()` Millisekunden in lokaler Zeit liefern
Script-Objekt Date
Integer 0 bis 999

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                        + "="
                        + escape(Value)
                        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                        + "="
                        + escape(Value)
                        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für Cookie_Name
    //      Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen

```



```

        var Gefunden=false;
        var Index = 0;
        do
        {
            // aktuelles Cookie lesen
            CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

            // und auf Name prüfen
            if (Name == CookieNameUndWertAlsFeld [0])
            {
                CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
                Gefunden=true;
            }
            else
            {Index++;}
        }
        while (      ( !Gefunden )
            && ( Index < AnzahlCookies)
            );

        return CookieWert;
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachennamen definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);
    }

```



```

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
            + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
            );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getMinutes()           Minuten in lokaler Zeit liefern
                        Script-Objekt Date
                        Integer 0 bis 59

Beispiel 1:
<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)    // Name und Wert sind Strings
    {
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
    }

    function LeseCookieWert(Name)
    // Name des zu lesenden Cookie ist String
    // liefert Cookiewert aus unescape() als String
    {
        var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert
    }

```



```

// Feld der Cookies referenzieren
// erzeugt durch Separation anhand Semikolon
var CookieFeld = document.cookie.split("; ");
var AnzahlCookies = CookieFeld.length;

// Feldelement umfasst Name und Wert des Cookie
// Aufbau Cookie_Name = Cookie_Wert
// Separator ist Gleichheitszeichen
// Index 0 für Cookie_Name
// Index 1 für Cookie_Wert
var CookieNameUndWertAlsFeld;

// CookieFeld auslesen und auf Cookie laut Name prüfen
var Gefunden=false;
var Index = 0;
do
{
    // aktuelles Cookie lesen
    CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

    // und auf Name prüfen
    if (Name == CookieNameUndWertAlsFeld [0])
    {
        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
        && ( Index < AnzahlCookies )
        );

return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachename festlegen
// es können diverse Cachennamen definiert und somit Versionen von Cache
// verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
// es können diverse Attribute definiert und somit Versionen von Input-Daten
// verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt

```



```

        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache save
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
            + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
        );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getMonth()    Monat in lokaler Zeit liefern
                Script-Objekt Date
                Integer 0 bis 11

```

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

```




```

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    //           Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies)
            );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachname festlegen
//     es können diverse Cachennamen definiert und somit Versionen von Cache
//     verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
//     es können diverse Attribute definiert und somit Versionen von Input-Daten
//     verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();

```



```

        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

.getNamedItem()

Zeiger auf Attribut liefern anhand des Attributnamen (analog zu ID oder NAME-Attribut)



ab IE 6.x

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        var ZeigerAufFeldElement = ZeigerAufFeld.getNamedItem("align");
        alert("ALIGN Attribut Wert = " + ZeigerAufFeldElement.value);
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P" ALIGN="center">Test</P>
</BODY>
</HTML>

```

.GetParentFolderName() Elternordner einer Datei liefern
 Datei muss nicht existieren

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
alert(DateiSystem.GetParentFolderName("../test\\text.txt"));

```

.getSeconds() Sekunden in lokaler Zeit liefern
 Script-Objekt Date
 Integer 0 bis 59

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)      // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =      Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)      // Name und Wert sind Strings
    {
        var document.cookie =      Name
                                + "="
                                + escape(Value)
                                + "; expires=Fri, 31 Dec 1999 23:59:59GMT"; // altes Datum
    }

    function LeseCookieWert(Name)
    // Name des zu lesenden Cookie ist String
    // liefert Cookiewert aus unescape() als String
    {
        var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

        // Feld der Cookies referenzieren
        // erzeugt durch Separation anhand Semikolon
        var CookieFeld = document.cookie.split("; ");
        var AnzahlCookies = CookieFeld.length;

        // Feldelement umfasst Name und Wert des Cookie
        // Aufbau      Cookie_Name = Cookie_Wert
        //                      Separator ist Gleichheitszeichen
        // Index 0 für Cookie_Name
        // Index 1 für Cookie_Wert
        var CookieNameUndWertAlsFeld;

        // CookieFeld auslesen und auf Cookie laut Name prüfen
        var Gefunden=false;
        var Index = 0;
        do
        {
            // aktuelles Cookie lesen
            CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

            // und auf Name prüfen
            if (Name == CookieNameUndWertAlsFeld [0])
            {

```



```

        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while (      ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachennamen definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = ZeitpunktJetzt.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"

```



```

        CLASS="user_data_speicher_klasse"
        TYPE="text"
    >
    <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
    <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
</FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n.... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

.GetSpecialFolder() Objekt FileSystemObject.Folder von Ordnern des Betriebssytemes erzeugen

Beispiel:

```

var DateiSystem        = new ActiveXObject("Scripting.FileSystemObject");
var Ordner             = DateiSystem.GetSpecialFolder(0);
var DateiOffen         = Ordner.CreateTextFile("test.txt");
DateiOffen.writeline("Test");
DateiOffen.close();

```

.GetTempName() Bezeichner anhand Zufallszahl erzeugen
Bezeichner verwendbar als Datei- oder Ordner-Bezeichner

Beispiel:

```

var DateiSystem        = new ActiveXObject("Scripting.FileSystemObject");
var Ordner             = DateiSystem.GetSpecialFolder(2);
var Dateiname          = DateiSystem.GetTempName();
var DateiOffen         = Ordner.CreateTextFile(Dateiname);
DateiOffen.writeline("Test");
DateiOffen.close();

```

.getTime() Zeitwert als Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit liefern
positiv oder negativ möglich (wenn < 0 so vor obigen Datum)
Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit
wenn < 0 so vor 01.01. 1970 0 Uhr Weltzeit

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)     // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =     Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)     // Name und Wert sind Strings
    {
        var document.cookie =     Name
                                + "="
                                + escape(Value)

```



```
+ "; expires=Fri, 31 Dec 1999 23:59:59GMT"; // altes Datum
```

```
function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für CookieName
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while ( ( !Gefunden )
        && ( Index < AnzahlCookies )
        );

    return CookieWert;
}
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    // es können diverse Cachennamen definiert und somit Versionen von Cache
    // verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    // es können diverse Attribute definiert und somit Versionen von Input-Daten
    // verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;
```



```

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getTimezoneOffset()    Minuten der lokalen Zeit als Differenz zur Weltzeit liefern, also das Offset der lokalen Zeitzone
                        Script-Objekt Date
                        Integer ab 0

```

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings

```




```

    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für CookieName
    //      Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies )
            );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachennamen definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

```



```

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;

    // Attribut instanzieren und mit Daten füllen
    InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

    // und Cache saveen
    InputDatenObjekt.save(FreierCacheName);
}

function InputLaden()
{
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
}

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n.... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }

```



</SCRIPT>

Beispiel 4:

```
var Jetzt = new Date();
alert(Jetzt.toLocaleString());
```

```
.getUTCDate()      Tag des Monats in Weltzeit liefern
                   Script-Objekt Date
                   Integer 1 bis 31
```

Beispiel 1:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    //           Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies)
            );

    return CookieWert;
}
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
```



```

<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachennamen definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag = 86400000;

```



```

var DatumDokumentErzeugung = new Date(document.fileCreatedDate);

var DatumHeute = new Date();

var TagesDifferenz = ( DatumHeute.getTime()
                      - DatumDokumentErzeugung.getTime()
                      )
                    / AnzahlMillisekundenProTag;

alert( "Dokument erzeugt am " + DatumDokumentErzeugung
      + "\n..... also vor " + parseInt(TagesDifferenz) + " Tagen"
      );
}
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getUTCDay()      Tag der Woche in Weltzeit liefern
                  Script-Objekt Date
                  Integer 0 bis 6
                   0 ist Sonntag
                   6 ist Samstag

```

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für CookieName
    //      Index 1 für CookieWert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld[0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld[1]);
            Gefunden=true;
        }
    }
    while (Index < AnzahlCookies);
}

```



```

    }
    else
    {Index++;}
}
while (      ( !Gefunden )
    && ( Index < AnzahlCookies)
);

return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachenames definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++++ Zeitstempel ++++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitpunktUTC = Zeitpunkt.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitpunktUTC;

        // ++++++++ Daten chachen ++++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"

```



```

        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

.getUTCFullYear() Jahreszahl vierstellig in Weltzeit liefern
 Script-Objekt Date
 Integer maximal **1970 + bzw. -285,616 Jahre**

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)    // Name und Wert sind Strings
    {
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
    }

    function LeseCookieWert(Name)
    // Name des zu lesenden Cookie ist String
    // liefert Cookiewert aus unescape() als String
    {
        var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

        // Feld der Cookies referenzieren
        // erzeugt durch Separation anhand Semikolon
        var CookieFeld = document.cookie.split("; ");
        var AnzahlCookies = CookieFeld.length;

        // Feldelement umfasst Name und Wert des Cookie
        // Aufbau    Cookie_Name = Cookie_Wert
        // Separator ist Gleichheitszeichen
        // Index 0 für Cookie_Name
        // Index 1 für Cookie_Wert
        var CookieNameUndWertAlsFeld;
    }

```




```

        // CookieFeld auslesen und auf Cookie laut Name prüfen
        var Gefunden=false;
        var Index = 0;
        do
        {
            // aktuelles Cookie lesen
            CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

            // und auf Name prüfen
            if (Name == CookieNameUndWertAlsFeld [0])
            {
                CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
                Gefunden=true;
            }
            else
            {Index++;}
        }
        while ( ( !Gefunden )
            && ( Index < AnzahlCookies)
        );

        return CookieWert;
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse { behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    // es können diverse Cachennamen definiert und somit Versionen von Cache
    // verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    // es können diverse Attribute definiert und somit Versionen von Input-Daten
    // verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
    }

```



```

        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz =    ( DatumHeute.getTime()
                                - DatumDokumentErzeugung.getTime()
                                )
                                / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
            + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
        );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getUTCHours()    Stunden in Weltzeit liefern
                  Script-Objekt Date
                  Integer 0 bis 23

```

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

    function LoescheCookie (Name, Value)    // Name und Wert sind Strings
    {
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
    }

    function LeseCookieWert(Name)
    // Name des zu lesenden Cookie ist String
    // liefert Cookiewert aus unescape() als String
    {
        var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert
    }

```



```

// Feld der Cookies referenzieren
// erzeugt durch Separation anhand Semikolon
var CookieFeld = document.cookie.split("; ");
var AnzahlCookies = CookieFeld.length;

// Feldelement umfasst Name und Wert des Cookie
// Aufbau Cookie_Name = Cookie_Wert
// Separator ist Gleichheitszeichen
// Index 0 für Cookie_Name
// Index 1 für Cookie_Wert
var CookieNameUndWertAlsFeld;

// CookieFeld auslesen und auf Cookie laut Name prüfen
var Gefunden=false;
var Index = 0;
do
{
    // aktuelles Cookie lesen
    CookieNameUndWertAlsFeld = CookieFeld[Index].split("=");

    // und auf Name prüfen
    if (Name == CookieNameUndWertAlsFeld [0])
    {
        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
        && ( Index < AnzahlCookies )
        );

return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachename festlegen
// es können diverse Cachennamen definiert und somit Versionen von Cache
// verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
// es können diverse Attribute definiert und somit Versionen von Input-Daten
// verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++

```



```

        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag =86400000;

        var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

        var DatumHeute =new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getUTCMilliseconds()    Millisekunden in Weltzeit liefern
                          Script-Objekt Date
                          Integer 0 bis 999

```

Beispiel 1:

```

<SCRIPT>
    function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
    {
        var date = new Date();
        var document.cookie =    Name
                                + "="
                                + escape(Value)
                                + "; expires=" + date.toGMTString(); // aktuelles Datum
    }

```



```

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =   Name
                        + "="
                        + escape(Value)
                        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau   Cookie_Name = Cookie_Wert
    //           Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (      ( !Gefunden )
        && ( Index < AnzahlCookies)
        );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
.user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
// Cachename festlegen
// es können diverse Cachennamen definiert und somit Versionen von Cache
// verwaltet werden
var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
// es können diverse Attribute definiert und somit Versionen von Input-Daten
// verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++++ Zeitstempel ++++++++

```



```

        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag=86400000;

        var DatumDokumentErzeugung=new Date(document.fileCreatedDate);

        var DatumHeute=new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
                               / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```



.getUTCMinutes() Minuten in Weltzeit liefern
 Script-Objekt Date
 Integer 0 bis 59

Beispiel 1:

```
<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                            + "="
                            + escape(Value)
                            + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    //           Separator ist Gleichheitszeichen
    // Index 0 für Cookieen_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies )
            );

    return CookieWert;
}
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    // es können diverse Cachennamen definiert und somit Versionen von Cache
    // verwaltet werden
```




```

var FreierCacheName = "InputCache";

// Cache-Attribut festlegen
//      es können diverse Attribute definiert und somit Versionen von Input-Daten
//      verwaltet werden
var FreiesCacheAttribut = "InputCacheAttribut";

// zu cachende Daten referenzieren
var InputDatenObjekt = ID_Formular.ID_Input;

function InputSichern()
{
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = Zeitstempel.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;

    // Attribut instanzieren und mit Daten füllen
    InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

    // und Cache saveen
    InputDatenObjekt.save(FreierCacheName);
}

function InputLaden()
{
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
}

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
    window.onload=fnInit;

    function fnInit()
    {
        var AnzahlMillisekundenProTag = 86400000;

        var DatumDokumentErzeugung = new Date(document.fileCreatedDate);

        var DatumHeute = new Date();

        var TagesDifferenz = ( DatumHeute.getTime()
                               - DatumDokumentErzeugung.getTime()
                             )
    }

```



```

        / AnzahlMillisekundenProTag;

        alert( "Dokument erzeugt am " + DatumDokumentErzeugung
              + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
              );
    }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getUTCMonth()    Monat in Weltzeit liefern
                  Script-Objekt Date
                  Integer 0 bis 11

```

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    //      erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    //      Aufbau    Cookie_Name = Cookie_Wert
    //      Separator ist Gleichheitszeichen
    //      Index 0 für Cookie_Name
    //      Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (    ( !Gefunden )
            && ( Index < AnzahlCookies)
            );

    return CookieWert;
}

```



```
    }
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //      es können diverse Cachenames definiert und somit Versionen von Cache
    //      verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //      es können diverse Attribute definiert und somit Versionen von Input-Daten
    //      verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++++ Zeitstempel ++++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++++ Daten chachen ++++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>
```

Beispiel 3:

```
<SCRIPT>
```



```

window.onload=fnInit;

function fnInit()
{
    var AnzahlMillisekundenProTag =86400000;

    var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

    var DatumHeute =new Date();

    var TagesDifferenz =    ( DatumHeute.getTime()
                            - DatumDokumentErzeugung.getTime()
                            )
                        / AnzahlMillisekundenProTag;

    alert( "Dokument erzeugt am " + DatumDokumentErzeugung
          + "\n.... also vor " +parseInt(TagesDifferenz) + " Tagen"
          );
}
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

```

.getUTCSeconds() Sekunden in Weltzeit liefern
 Script-Objekt Date
 Integer 0 bis 59

Beispiel 1:

```

<SCRIPT>
function ErzeugeCookie(Name, Value)    // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value)    // Name und Wert sind Strings
{
    var document.cookie =    Name
                          + "="
                          + escape(Value)
                          + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau    Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookien_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
    }
}

```



```

        if (Name == CookieNameUndWertAlsFeld [0])
        {
            CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
            Gefunden=true;
        }
        else
        {Index++;}
    }
    while (      ( !Gefunden )
        && ( Index < AnzahlCookies)
    );

    return CookieWert;
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    // es können diverse Cachennamen definiert und somit Versionen von Cache
    // verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    // es können diverse Attribute definiert und somit Versionen von Input-Daten
    // verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>

```



```

<FORM ID="ID_Formular">
  <INPUT ID="ID_Input"
    CLASS="user_data_speicher_klasse"
    TYPE="text"
  >
  <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
  <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
</FORM>
</BODY>
</HTML>

```

Beispiel 3:

```

<SCRIPT>
  window.onload=fnInit;

  function fnInit()
  {
    var AnzahlMillisekundenProTag =86400000;

    var DatumDokumentErzeugung =new Date(document.fileCreatedDate);

    var DatumHeute =new Date();

    var TagesDifferenz =  ( DatumHeute.getTime()
                          - DatumDokumentErzeugung.getTime()
                          )
                        / AnzahlMillisekundenProTag;

    alert( "Dokument erzeugt am " + DatumDokumentErzeugung
          + "\n..... also vor " +parseInt(TagesDifferenz) + " Tagen"
          );
  }
</SCRIPT>

```

Beispiel 4:

```

var Jetzt = new Date();
alert(Jetzt.toLocaleString());

.getYear()          deprecated

.go()               Aktivierung irgendeiner Url aus dem Verlauf
                   entspricht beliebiger Selektion aus der Verlaufsliste
< 0
-1                  entspricht wie wenn nicht kodiert
                   entspricht Druck des Back-Buttons
                   je kleiner umso weiter zuvorliegend
> 0                 +1      entspricht Druck des Forward-Buttons
                   je größer umso weiter nachliegend
0                   entspricht Reload des aktuellen Dokumentes

```

Beispiel: Reload des Dokumentes nach Resize des Browserfensters

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
  function neuLaden()
  {
    if (document.all)
    {history.go(0);}
  }
// -->
</SCRIPT>
</HEAD>
<BODY onResize=neuLaden();">
</BODY>

```

```

.hasChildNodes()    prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten
                   (Textelemente) in einem Objekt
                   DOM nicht geändert

```

Beispiel:

```

<HTML>
<BODY onload="alert(ID_Div.hasChildNodes());">

```



```
<DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>
```

<code>.hasFeature()</code>	Objektzugehörigkeit zum DOM (Document Object Model-Standard) (HTML-DOM bzw. XML-DOM)
----------------------------	--

Beispiel:

```
var Wert = document.implementation.hasFeature("HTML", "1.0");  
// Objekt document prüfen ob im HTML-DOM Version 1.0 enthalten ist
```

.hasFocus()	Objekt im Focus-Zustand
true	Objekt hat den Focus
false	Objekt hat keinen Focus

<code>.hasOwnProperty()</code>	<p>prüfen ob zum Objekt eine Eigenschaft vorhanden ist, jedoch leider nicht aus Prototyping einer per new erzeugten Instanz des JScript-Objektes</p> <p>Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider nicht die Eigenschaft <code>.prototype</code> erzeugt ! Prototyping kann also nur innerhalb des Konstruktors erfolgen und nicht nachträglich per Eigenschaft <code>.prototype</code> .</p> <p>siehe auch <code>.prototype</code></p>
--------------------------------	--

Beispiel:

```
function MaximumErmittleIn ( )
{
    var max = this[0];    // this referenziert die Objekt-Instanz, in dem die Methode vorhanden ist,
                          // also Variable Feld

    for (var i = 1; i < this.length; i++)
    {
        if (max < this[i])
        { max = this[i]; }
    }

    return max;
}
```

```
// neue Methode per Prototyping hinzufügen zum JScript-Objekt Array
Array.prototype.NeuArrayMethode = MaximumErmitteln; // ohne () kodieren !
```

```
// Feld vom Array-Typ erzeugen mit der Methode .NeueArrayMethode()
var Feld = new Array(1, 2, 3, 4, 5, 6);      // 6 numerische Elemente
```

```
alert( (Feld.prototype.hasOwnProperty("NeueArrayMethode")); // leider false
alert( (Array.prototype.hasOwnProperty("NeueArrayMethode")); // true
```

`.hide()` ein angezeigtes Popup-Fenster schliessen
Hinweis: Das Popup-Fenster wird automatisch geschlossen, wenn ein anderes Objekt den Focus erhält bzw. aktiv wird, z.B. durch Klick des Users außerhalb des Popupfensters.
ändert Wert der Eigenschaft `.isOpen`

Beispiel für diverse Meldungsfenster per **jeweiligem** PopUp-Fenster (es kann immer nur genau 1 Popup-Fenster angezeigt werden):

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
```

```
var PopUpFensterFeld = new Array();
```

```
// nachfolgende Felder beschreiben die PopUp-Fenster und müssen
// identische Anzahl der Feldelemente haben
// Feld-Index ist die Nummer des PopUp-Fensters
```

```
var PopUpFenster_RahmenStyle_Feld = new Array
(
    "solid black 4px",
    "solid blue 3px",
    "solid green 2px"
);
```

```
var PopUpFenster_HintergrundFarbe_Feld = new Array  
(  
    "yellow",  
    "gray",
```



```

        "orange"
    );

    var PopUpFenster_Inhalt_Feld = new Array
    (
        "<B>Inhalt PopUpFenster 0<B>",
        "Inhalt PopUpFenster 1",
        "<TT>Inhalt PopUpFenster 2<TT>"
    );

    var PopUpFenster_X_Koordinate_Feld = new Array
    (
        100,
        150,
        200
    );

    var PopUpFenster_Y_Koordinate_Feld = new Array
    (
        150,
        200,
        250
    );

    var PopUpFenster_Breite_Koordinate_Feld = new Array
    (
        80,
        140,
        200
    );

    var PopUpFenster_Hoehe_Koordinate_Feld = new Array
    (
        100,
        140,
        180
    );

    var AnzahlPopUpFenster = PopUpFenster_Hoehe_Koordinate_Feld.length;

    function PopupFensterInstanzieren(NummerDesPopUpFensters, ZeigerAufElternFenster)
        // NummerDesPopUpFensters ab 0
    {
        PopUpFensterFeld[NummerDesPopUpFensters] =
            ZeigerAufElternFenster.createPopup();
    }

    function PopupFensterFuellen(NummerDesPopUpFensters)
    {
        // Body des Popup-Fensters gestalten
        var PopupFenster_Body =
            PopUpFensterFeld[NummerDesPopUpFensters].document.body;

        PopupFenster_Body.style.backgroundColor =
            PopUpFenster_HintergrundFarbe_Feld[NummerDesPopUpFensters];

        PopupFenster_Body.style.border =
            PopUpFenster_RahmenStyle_Feld[NummerDesPopUpFensters];

        PopupFenster_Body.innerHTML =
            PopUpFenster_Inhalt_Feld[NummerDesPopUpFensters];
    }

    function PopupFensterAnzeigen(NummerDesPopUpFensters,
        ObjektZuDemPopUpFensterRelativPositioniertIst
    )
    {
        // Popup-Fenster anzeigen und damit öffnen
        PopUpFensterFeld[NummerDesPopUpFensters].show(
            PopUpFenster_X_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Y_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Breite_Koordinate_Feld[NummerDesPopUpFensters],
            PopUpFenster_Hoehe_Koordinate_Feld[NummerDesPopUpFensters],
            ObjektZuDemPopUpFensterRelativPositioniertIst
        );
    }

```




```

    }
    );

    function PopUpFensterSchliessen(NummerDesPopUpFensters)
    {
        var Zeiger = PopUpFensterFeld[NummerDesPopUpFensters];

        if (Zeiger.isOpen)
        { Zeiger.hide(); }
    }

    function Init()
    {
        // PopUpFenster instanzieren im aktuellen Fenster (window)
        for (var i = 0 ; i < AnzahlPopUpFenster; i++)
        {
            PopupFensterInstanzieren(i, window);
            PopupFensterFuellen(i);
        }
    }
</SCRIPT>
</HEAD>
<BODY onload="Init();">
    Durch Klick ausserhalb des Popup-Fensters wird dieses auch automatisch geschlossen.
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 anzeigen"
        onclick=" PopupFensterAnzeigen(0, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 anzeigen"
        onclick=" PopupFensterAnzeigen(1, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 anzeigen"
        onclick=" PopupFensterAnzeigen(2, document.body);"
    >
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 schliessen"
        onclick=" PopUpFensterSchliessen(0);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 schliessen"
        onclick=" PopUpFensterSchliessen(1);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 schliessen"
        onclick=" PopUpFensterSchliessen(2);"
    >
</BODY>
</HTML>

```

.ImportExportFavorites() Import und Export der Favoritenliste aus dem/ in das Netscape-Bookmark-Format
 User muss Import/Export bestätigen per Dialogbox
 es wird HTTP benutzt
 Ziel-bzw. Quellort: Es wird durch Import immer hinzugefügt
 Es wird nach Export nicht gelöscht.
 siehe Objekt window.external

Beispiel:

```
window.external.ImportExportFavorites(true,"http://www.test.com");
```

.indexOf() Suche einer Teilkette in einem String bzw. Stringliteral und die Startposition der Teilkette als Index
 im String bzw. Stringliteral liefern
 es wird nur das ERSTE Auffinden der Teilkette geliefert
 Suche im String erfolgt von links nach rechts
 unterscheidet Gross und Klein
 siehe Script-Objekt String

Beispiel 1:

```

var StringLiteral ="StringLiteral";
StringLiteral.indexOf("gLi", 2)    liefert 5
StringLiteral.indexOf("gLi")       liefert 5
StringLiteral.indexOf("gLi", 10)   liefert -1

```



"StringLiteral".indexOf("gLi", 2) liefert 5

Beispiel 2 E-Mail-Adresse auf "@" prüfen:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function pruefe_zeichenkette(zeichenkette)
    {
        if (zeichenkette.indexOf('@') = -1)
        {alert("@ fehlt !");}
        else
        {alert("@ gefunden !");}
    }
//-->
</SCRIPT>
</HEAD>

<BODY>
<FORM>
    e-Mail:
    <INPUT TYPE="text"
        NAME="Mail"
        VALUE=""
    >
    <INPUT TYPE="button"
        VALUE="Ueberpruefen"
        onclick=" pruefe_zeichenkette (this.form.Mail.value)"
    >
</FORM>
</BODY>
</HTML>
```

Beispiel 3: Numerischen Wert als Zeichenkette liefern und dabei Dezimalpunkt zu Dezimalkomma umwandeln

Es wird angenommen, dass genau ein Dezimalpunkt vorkommt.

```
function punkt_zu_komma (numerischer_wert)
{
    var zeichenkette = numerischer_wert.toString();
    var funktionswert=zeichenkette;

    var pos_punkt = zeichenkette.indexOf(".");

    if(pos_punkt >=0)
    {
        funktionswert =      zeichenkette.substring(0, pos_punkt)
                           + ","
                           + zeichenkette.substring(pos_punkt + 1, zeichenkette.length);
    }

    return zeichenkette;
}
```

.inRange() prüfen ob 2 Textbereiche sich einschliessen z.B. bei verschachtelten DIV's
per textrange Objekt
nur unter Windows 32-Bit

Beispiel:

```
<HTML>
<SCRIPT>
    window.onload=fnCheck;

    function fnCheck()
    {
        var TextBereich = document.body.createTextRange();

        var TextBereichKopie = TextBereich.duplicate();

        alert(TextBereich.inRange(TextBereichKopie));
    }
</SCRIPT>
<BODY>
    <DIV>
```



```

        Test
    </DIV>
</BODY>
</HTML>

```

.insertAdjacentElement() Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert

Beispiel:

```

<SCRIPT>
function Hinzufuegen()
{
    var ZeigerAufLI = document.createElement("LI");
    ID_Liste.children(0).insertAdjacentElement("beforeBegin", ZeigerAufLI);
    ZeigerAufLI.innerText = "Listeneintrag 0";
}
</SCRIPT>
<BODY>
    <OL ID = "ID_Liste">
        <LI>Listeneintrag 1</LI>
        <LI>Listeneintrag 2</LI>
        <LI>Listeneintrag 3</LI>
    </OL>
    <INPUT TYPE = "button" VALUE = " Hinzufuegen" onclick=" Hinzufuegen()">
</BODY>

```

.insertAdjacentHTML() HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen **nicht** ausgeführt eingefügter Code wird **nur** dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert

Beispiel:

```

var HTML_Code =    "<INPUT TYPE =button"
                  +    " VALUE='Bitte klicken'"
                  +    " onclick='Anzeige()'"
                  + ">"
                  + "<BR>";

var JavaScript_Code =    "<SCRIPT DEFER>"           // DEFER muss kodiert sein
                        + "function Anzeige(){alert('Anzeige aktiv') }"
                        + "</SCRIPT>";

ID_Div.insertAdjacentHTML("afterBegin", HTML_Code + JavaScript_Code);

<DIV ID="ID_Div">
    Test
</DIV>

```

.insertAdjacentText() Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert

Beispiel:

```

var PlainText = " Hinzugefuegter Text ";
ID_Div.insertAdjacentText("afterBegin", PlainText);

<DIV ID="ID_Div">
    Test
</DIV>

```

.insertBefore() Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird geändert

Beispiel:

```

<SCRIPT>
function Einfuegen()
{
    var ZeigerAufNeuesLI = document.createElement("LI");

```



```

        ID_UL.insertBefore(ZeigerAufNeuesLI, ID_LI);
        ZeigerAufNeuesLI.innerText="2";
    }
</SCRIPT>
</HEAD>
<BODY>
        <SPAN onclick= Einfuegen()>Klick</SPAN>
        <UL ID="ID_UL">
            <LI >1</LI>
            <LI ID="ID_LI">3</LI>
            <LI >4</LI>
        </UL>
</BODY>

```

.insertCell() Zelle TD in die Zeile einer Tabelle einfügen
 Zelle TH **nicht direkt** erzeugbar: Es muss .innerHTML mit -Tag gefüllt werden
 siehe Objekt table.tr

Beispiel :

```

var Zeile = zeiger_auf_tabelle.insertRow();
var Zelle = Zeile.insertCell();
Zelle.innerHTML = " <I>Test </I>";

```

.insertData() Teilkette in ein Objekt einfügen

.insertRow() Zeile in die Tabelle einfügen
 siehe Objekt table
 siehe Objekt table.tBody
 siehe Objekt table.tHead
 siehe Objekt table.tFoot

Beispiel :

```

var Zeile = zeiger_auf_tabelle.insertRow();
var Zelle = Zeile.insertCell();
Zelle.innerHTML = " <I>Test </I>";

```

.isComponentInstalled() Verfügbarkeit der Komponente (gedownloadet UND installiert UND aktuelle Versionsnummer
 ist mindestens die minimale Versionsnummer)
 Behavior .style.clientCaps

.isEqual() Auf Identität zweier Textbereiche prüfen
 per textrange Objekt

isFinite() ermittelt, ob Wert einer Instanz numerisch endlich ist

.isHomePage() Lage der Homepage auf einer Domain
 true ist eine Homepage UND Homepage liegt auf gleicher Domain
 false ist keine Homepage und/oder Homepage liegt auf anderen Domain

isNaN() ermittelt, ob Wert einer Instanz nicht numerisch ist

Beispiele:

```

var Kette ="10.23";
var Wert =10.23;

var Ergebnis1=parseFloat(Kette);
var Ergebnis2=floatValue=parseFloat(Wert);

alert(isNaN(Ergebnis1);
alert(isNaN(Ergebnis2);

```

Hinweis: Methoden parseFloat und parseInt liefern NaN, wenn Parameter nicht numerisch ist

Bsp: var zahl=parseFloat("text"); ifNaN() liefert true

.isPrototypeOf() prüfen ob Objekt per new erzeugt wurde, also eine Instanz eines anderen JScript-Objektes ist
 Für private Objekte, die per new-Anweisung mit privatem Konstruktor erzeugt wurden, wird leider **nicht**
 die Eigenschaft .prototype erzeugt ! Prototyping kann also nur innerhalb des Konstruktors
 erfolgen und nicht nachträglich per Eigenschaft .prototype .
 siehe auch .prototype

Beispiel:

```

var Variable = new RegExp();
alert( (RegExp.prototype.isPrototypeOf(Variable));                      // true.

```

.IsSubscribed() auf Verfügbarkeit der Unterschrift zu einer Microsoft Active Channel-Datei prüfen
 (Channel Definition Format-Datei mit Dateisuffix ist *.cdf)
 nur für Dateien in gemeinsamer Domain, also nicht domain-übergreifend
 liefert immer Scriptfehler, wenn Domain-Wechsel durch Url der CDF-Datei erzeugt wird



siehe Objekt window.external

.italics() HTML-Tag <I> erzeugen in der Form <I>text</I>
siehe Script-Objekt String

Beispiel:

```
var StringLiteral    ="Text der I wird"
var HTML_Kette      = StringLiteral.italics();
HTML_Kette          = "Text der I wird".italics();
```

entspricht <I>Text der I wird</I>

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !

Beispiel 1:

```
<SCRIPT LANGUAGE="JScript">
var ZeigerAufCollectionDocumentAll = document.all;

if (ZeigerAufObjekt!=null)
{
    for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
    {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
}
</SCRIPT>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
function Init()
{
    var ZeigerAufFeld = ID_P.attributes;
    for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
        // hier numerischer Index
        var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
        // true oder false
        var KnotenName = ZeigerAufFeldElement.nodeName;
        // String
        var KnotenWert = ZeigerAufFeldElement.nodeValue;
        alert(
            "Knotenname = "
            + KnotenName
            + " mit spezifiziert = "
            + AttributWertSpezifiziert
            + " und Wert = "
            + KnotenWert
        );
    }
}
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
<P ID="ID_P">Test</P>
</BODY>
</HTML>
```

.item() Element einer Collection liefern laut aktueller Position in der Collection
zur Positionierung innerhalb der Collection wird ein interner Index verwendet
siehe Enumerator JScript-Objekt

Beispiel Laufwerke auf dem PC des Users ermitteln:

```
var DateiSystem      = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner
```

```
for ( ; ! DateiSystem_Laufwerke.atEnd();DateiSystem_Laufwerke.moveNext())
```



```

{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";
}
alert (Kette);

```

.Item() Inhalt des Datenfeldes anhand Schlüssel liefern

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Date anzeigen
    alert(DatenSpeicher.Item(DatenSchluessel));
}

```

.Item() Eintrag in der Collection FileSystemObject.Drives liefern
 nur in Verbindung mit JScript-Objekt Enumerator

Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netiname oder Volume-Bezeichner

for ( ; ! DateiSystem_Laufwerke.atEnd(); DateiSystem_Laufwerke.moveNext() )
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;

```




```

var DatenReferenz = DatenSpeicher.Items();

// und Feld bilden
//     VisualBasic-Feld erzeugen
var DatenOhneSchluessel_Feld = new VBArray(DatenReferenz);
//     und nach JScript-Feld konvertieren
DatenOhneSchluessel_Feld = DatenOhneSchluessel_Feld.toArray();

// und Daten anzeigen
var DatenOhneSchluessel_FeldLaenge = DatenOhneSchluessel_Feld.length

if (DatenOhneSchluessel_FeldLaenge > 0)
{
    for (var i = 0 ; i < DatenOhneSchluessel_FeldLaenge; i++)
        {alert("Date " + i + " = " + DatenOhneSchluessel_Feld [i]);}
}
else
{alert("Dictionary ist leer!");}
}

```

`.javaEnabled()` prüfen auf generelle Ausführbarkeit von Javacode, also ob Java-Maschine im Browser aktiv ist
per navigator Objekt
true Javacode ist ausführbar
false Javacode ist nicht ausführbar

Hinweis: Eigenschaft `.javaEnabled` des Behavior `clientCaps` zeigt die Verfügbarkeit der
Microsoft Virtuellen Maschine für Java an

Achtung: Aufgrund eines Streites mit dem Unternehmen Sun entwickelt Microsoft ab IE 6.x die MS VM nicht mehr weiter, hält die VM aber noch per Download verfügbar. Für eine aktuellere Version muss der User selbständig beim Unternehmen Sun eine Lizenz für Java erwerben und die zugehörige Software installieren (inklusive Updates). Es reicht dabei die Run-Time-Version der Virtuellen Java Maschine von Sun. Ob allerdings Sun-Java-Standards in Microsoft-Produkten implementiert werden, ist nicht gesichert.

`.javaEnabled()` Java-Verfügbarkeit
ab IE 6.x
siehe Objekt `window.clientInformation`
true Javacode ist ausführbar
false Javacode ist nicht ausführbar

Hinweis: Eigenschaft `.javaEnabled` des Behavior `clientCaps` zeigt die Verfügbarkeit der
Microsoft Virtuellen Maschine für Java an

Achtung: Aufgrund eines Streites mit dem Unternehmen Sun entwickelt Microsoft ab IE 6.x die MS VM nicht mehr weiter, hält die VM aber noch per Download verfügbar. Für eine aktuellere Version muss der User selbständig beim Unternehmen Sun eine Lizenz für Java erwerben und die zugehörige Software installieren (inklusive Updates). Es reicht dabei die Run-Time-Version der Virtuellen Java Maschine von Sun. Ob allerdings Sun-Java-Standards in Microsoft-Produkten implementiert werden, ist nicht gesichert.

`.join()` Feld auslesen und als String liefern, wobei jedes Feldelement mit einem freidefinierbaren
Trenner im String getrennt wird, der mit eingebaut wird
Feld elementweise nach String kopieren und als 1 gemeinsamen String liefern
Feld hat die Objektklasse `Script-Objekt Array`

Beispiele:

```

var Feld = new Array(0,1,2,3,4);
alert (Feld.join("-")); // "0-1-2-3-4"
alert(Feld.join());    // "0,1,2,3,4"

var Feld = new Array("Wind","Rain","Fire");
var Kette = Feld.join(" + ");    // Kette enthält "Wind + Rain + Fire"

```

`.Key()` Datenschlüssel im Dictionary ändern

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

```




```
// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchlüssel, Date);

    // dann Schlüssel ändern
    DatenSpeicher.Key(DatenSchlüssel) = "b";

    // und Meldung ob Date vorhanden ist
    alert(SchlüsselVorhanden("b"));
}
```

.Keys() Inhalt aller Schlüsselfelder des Dictionary-Objektes als Referenz liefern, die als Zeiger für den Konstruktor `new VBArray ()` (VisualBasic-Anweisung) dient
Schlüsselfelder-Reihenfolge laut Folge im Dictionary

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden (DatenSchlüssel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchlüssel, Date);

    // und Daten-Referenz bilden
    var SchlüsselReferenz = DatenSpeicher.Keys();

    // und Feld bilden
    // VisualBasic-Feld erzeugen
    var SchlüsselOhneDaten_Feld = new VBArray(SchlüsselReferenz);
    // und nach JScript-Feld konvertieren
    SchlüsselOhneDaten_Feld = SchlüsselOhneDaten_Feld.toArray();

    // und Daten anzeigen
    var SchlüsselOhneDaten_FeldLaenge = SchlüsselOhneDaten_Feld.length

    if (SchlüsselOhneDaten_FeldLaenge > 0)
    {
        for (var i = 0 ; i < SchlüsselOhneDaten_FeldLaenge; i++)
        {alert("Schlüssel " + i + " = " + SchlüsselOhneDaten_Feld [i]);}
    }
    else
    {alert("Dictionary ist leer!");}
}
```

.lastIndexOf() Suche einer Teilkette in einem String bzw. Stringliteral und die Startposition der Teilkette als Index im String bzw. Stringliteral liefern
es wird nur das LETZTE Auffinden der Teilkette geliefert
Suche im String erfolgt von links nach rechts
unterscheidet Gross und Klein
siehe Script-Objekt String

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.lastIndexOf("t", 2) liefert 8
StringLiteral.lastIndexOf("t") liefert 8
StringLiteral.lastIndexOf("t", 10) liefert -1
"StringLiteral".indexOf("t", 2) liefert 8
```

.lastPage() letzte Seite des Datasets in Tabelle anzeigen
Eigenschaft `.dataPageSize` muss belegt worden sein
siehe Objekt table

Beispiel:

Datensätze liegen in der Datei adress.txt
Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
hat folgenden Inhalt:



1. Satz
ab 2. Satz

vorname;name;telefon
Guericke;"Otto, von";0815
Willmann;Theo;1234
Xantippe;Isa;5678
Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) { Kette = "+"; }

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            { ID_Datenbank.recordset.MoveNext(); }
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        { alert("Letzter Datensatz erreicht!"); }
    }

    function rueckwaerts(AnzahlSaetze)
    {
        // vorhergehende Seite anzeigen
        document.all.ID_Tabelle.previousPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition > 1)
            { ID_Datenbank.recordset.MovePrevious(); }
        }

        if (ID_Datenbank.recordset.AbsolutePosition == 1)
        { alert("Erster Datensatz erreicht!"); }
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
```



```

>
    <THEAD>
    <TR>
        <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
        <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
        <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
    </TR>
    </THEAD>
    <TBODY>
    <TR>
        <TD><DIV DATAFLD="vorname"></DIV></TD>
        <TD><DIV DATAFLD="name"></DIV></TD>
        <TD><DIV DATAFLD="telefon"></DIV></TD>
    </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT          TYPE="button"
                VALUE="Zurueck"
                onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->

<INPUT          TYPE="button"
                VALUE="Vorwaerts"
                onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->
</BODY>
</HTML>

```

`.link()` HTML-Link <A> als HTML-Kette erzeugen in der Form "text"
und ohne weitere Attribute ID, HREF etc.
siehe Script-Objekt String

Beispiel:

```

var StringLiteral    ="Sichtbarer Ankertext"
var HTML_Kette       = StringLiteral.link("WertDesHREFAttributes");
HTML_Kette           = "Sichtbarer Ankertext".link("WertDesHREFAttributes");

```

entspricht Sichtbarer Ankertext

`eval(HTML_Kette);` erzeugt Fehler, da HTML-Code von `eval` nicht ausgeführt werden kann
`document.write(HTML_Kette);` zeigt den Anker an und erzeugt Eintrag in der Collection `document.anchors`

`.load()` gespeicherten `UserDataStore` (User-Cache) auslesen per `.style.userData Behavior`
Cache muss per Methode `.save()` zum Behavior gespeichert worden sein

Beispiel:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    //     es können diverse Cachenames definiert und somit Versionen von Cache
    //     verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    //     es können diverse Attribute definiert und somit Versionen von Input-Daten
    //     verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitpunkt = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen

```



```

        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;

        // ++++++ Daten chachen ++++++
        // aktuelle Daten holen laut Input-Objekt
        var InputDaten = InputDatenObjekt.value;

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

.localeCompare() Vergleich zweier Strings oder Stringlitterale bezüglich ihrer Sortierfolge laut aktuelle Vorgaben zur Sortierungsfolge auf dem PC des Users
nur IE ab 5.5
siehe Script-Objekt String

.log() liefert den Logarithmus zur Basis E der zahl (natürlicher Logarithmus)
siehe Script-Objekt Math

.match() Suche in einem String oder String-literal per RegExp-Objekt (siehe dort Methode .exec())
siehe Script-Objekt String

Beispiel 1:

```

var Kette           = "The rain in Spain falls mainly in the plain";
var RegExpressionAusdruck = /ain/i;
var Feld1           = Kette.match(RegExpressionAusdruck);
var Feld2           = "The rain in Spain falls mainly in the plain".match(RegExpressionAusdruck);

```

Beispiel 2:

```

var zu_durchsuchende_kette = "Otto";
var such_muster            = /(wOtto)/g; // vom Typ RegExp, also regulärer Ausdruck
var ergebnis_feld         = zu_durchsuchende_kette.match(such_muster);

```

.max() größte Zahl aller Zahlen liefern
siehe Script-Objekt Math

.mergeAttributes() alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
Attribute sind: HTML
Events
Styles
ab IE 5.01 auch ID, NAME
Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
DOM wird geändert

Beispiel:

```

<SCRIPT>
function Mischen()
{ ID_SPAN.children[1].mergeAttributes(ID_SPAN.children[0]);}

```



```

</SCRIPT>
<SPAN ID=ID_SPAN>
  <DIV ID="ID_Div_Quelle"
    ATTRIBUTE1="true"
    ATTRIBUTE2="true"
    onclick="alert('click');"
    onmouseover="this.style.color='#0000FF';"
    onmouseout="this.style.color='#000000';"
  >
    Quell<B>Div</B>
  </DIV>
  <DIV ID="ID_Div_Ziel">
    Ziel-Div
  </DIV>
</SPAN>

<INPUT TYPE="button" VALUE=" Mischen" onclick="Mischen()">

```

.min() kleinste Zahl aller Zahlen liefern
siehe Script-Objekt Math

.move() Textbereich-Zeichen-Zeiger bewegen bezüglich aktueller Position im Textbereich
per textrange Objekt
nur unter Windows 32-Bit

.Move() vorhandene Datei verschieben (sonst Fehler erzeugt)

Beispiel:

```

var DateinameMitPfad      = "c:\\test.txt";
var Dateisystem            = new ActiveXObject("Scripting.FileSystemObject");
var Datei                 = Dateisystem.GetFiles(DateinameMitPfad);
Datei.Move("c:\\windows\\desktop\\");

```

.Move() vorhandenen Ordner verschieben (sonst Fehler erzeugt)

Beispiel:

```

var OrdnerName            = "c:\\windows\\desktop\\";
var Dateisystem           = new ActiveXObject("Scripting.FileSystemObject");
var Ordner                = Dateisystem.GetFolder(OrdnerName);
Ordner.Move("d:\\recycled\\");

```

.moveBy() linke obere Fenster-Ecke um Pixeldifferenz auf dem Bildschirm verschieben
Fenster ganz aus dem BS verschieben ist möglich
Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Bildschirms
nicht anwendbar bei Dialog-Fenster:
Dafür dialogHeight, dialogWidth, dialogTop und dialogLeft verwenden.
siehe Objekt window

Beispiel für Fenster des Browser rausschieben und danach neu anzeigen:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
var BrowserFenster_AktuelleBreite=2000; // sollte größer als max. übliche Auflösung sein
var BrowserFenster_AktuelleHoehe=2000; // sollte größer als max. übliche Auflösung sein

function moveWin()
{
    for (var i = 1; i < BrowserFenster_AktuelleBreite; i++)
    {window.moveBy(1, 1);}

    window.moveBy((-1)* BrowserFenster_AktuelleBreite,(-1) * BrowserFenster_AktuelleHoehe);
}
-->
</SCRIPT>
</HEAD>
<BODY onload="moveWin();">
</BODY>
</HTML>

```

.moveEnd() Textbereich-Ende neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich
per textrange Objekt
nur unter Windows 32-Bit

.MoveFile(en) Datei(en) verschieben



Wenn Move abbricht, so bleiben bisher erfolgte Verschiebungen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.MoveFile("c:\\eigene dateien\\*.doc", "c:\\recycled\\")
```

.moveFirst()

aktuelle Position innerhalb der Collection auf das erste Element der Collection setzen (auch wenn Collection leer ist)
zur Positionierung innerhalb der Collection wird ein interner Index verwendet
Element an aktueller Position ermitteln per .item()
siehe Enumerator JScript-Objekt

Beispiel Laufwerke auf dem PC des Users ermitteln:

```
var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

// erstes Laufwerk im Dateisystem einstellen (erstes Element im Enumerator-Objekt)
DateiSystem_Laufwerke.moveFirst();
do
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";

    // nächstes Laufwerk positionieren (nächstes Element im Enumerator-Objekt)
    DateiSystem_Laufwerke.moveNext();
}
while (!DateiSystem_Laufwerke.atEnd());

alert(Kette);
```

.MoveFolder()

Ordner verschieben
Wenn Move abbricht, so bleiben bisher erfolgte Verschiebungen leider erhalten

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
DateiSystem.MoveFolder("c:\\eigene dateien\\", "c:\\recycled\\")
```

.moveNext()

aktuelle Position innerhalb der Collection auf das nächste Element der Collection setzen (auch wenn Collection leer ist)
zur Positionierung innerhalb der Collection wird ein interner Index verwendet
Element an aktueller Position ermitteln per .item()
siehe Enumerator JScript-Objekt



Beispiel Laufwerke auf dem PC des Users ermitteln:

```

var DateiSystem          = new ActiveXObject("Scripting.FileSystemObject");
var DateiSystem_Laufwerke = new Enumerator(DateiSystem.Drives);
var Kette = "";
var Laufwerk;
var LaufwerkBuchstabe;
var LaufwerkName; // Netzname oder Volume-Bezeichner

// erstes Laufwerk im Dateisystem einstellen (erstes Element im Enumerator-Objekt)
DateiSystem_Laufwerke.moveFirst();
do
{
    // Laufwerk ermitteln
    Laufwerk = DateiSystem_Laufwerke.item();

    // Laufwerksbuchstabe ermitteln
    LaufwerkBuchstabe = Laufwerk.DriveLetter;
    Kette = Kette + LaufwerkBuchstabe + " - ";

    // Art des Laufwerkes ermitteln

    if (Laufwerk.DriveType == 3)
    {
        // ist Netzlaufwerk

        // öffentlicher Netz-Name des Laufwerkes
        LaufwerkName = Laufwerk.ShareName;
    }
    else
    {
        // nicht Netzwerk-Laufwerk

        // prüfen ob Laufwerk bereit ist
        if (Laufwerk.IsReady)
        {
            // ist bereit, dann Volume-Bezeichner ermittelbar
            LaufwerkName = Laufwerk.VolumeName;
        }
        else
        { LaufwerkName = "[Drive not ready]"; }
    }

    Kette += LaufwerkName + "\n";

    // nächstes Laufwerk positionieren (nächstes Element im Enumerator-Objekt)
    DateiSystem_Laufwerke.moveNext();
}
while (!DateiSystem_Laufwerke.atEnd());

alert(Kette);

```

.moveRow() 2 Zeilen in der Tabelle austauschen
 siehe Objekt table
 siehe Objekt table.tBody
 siehe Objekt table.tHead
 siehe Objekt table.tFoot

Beispiel:

```

<TABLE ID="ID_Tabelle" BORDER CELSPACING=10>
  <TR>
    <TD>Body Zeile 1 Zelle 1</TD>
    <TD>Body Zeile 1 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 2 Zelle 1</TD>
    <TD>Body Zeile 2 Zelle 2</TD>
  </TR>
  <TR>
    <TD>Body Zeile 3 Zelle 1</TD>
    <TD>Body Zeile 3 Zelle 2</TD>
  </TR>
</TABLE>
<BUTTON onclick="ID_Tabelle.moveRow(0,1);">verschieben Zeile 0 auf Zeile 1</BUTTON>

```



<code>.moveStart()</code>	Textbereich-Anfang neu setzen (für den Textbereich-Zeiger) bezüglich aktueller Position im Textbereich per <code>textrange</code> Objekt nur unter Windows 32-Bit
<code>.moveTo()</code>	linke obere Fenster-Ecke laut Pixel-Position auf dem Bildschirm positionieren Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Bildschirms nicht anwendbar bei Dialog-Fenster: Dafür <code>dialogHeight</code> , <code>dialogWidth</code> , <code>dialogTop</code> und <code>dialogLeft</code> verwenden. siehe Objekt <code>window</code>
<code>.moveToBookmark()</code>	Textbereich-Zeichen-Zeiger auf eine Textmarke (Bookmark) im Textbereich positionieren Textmarke wurde gesetzt per Methode <code>.getBookmark()</code> per <code>textrange</code> Objekt nur unter Windows 32-Bit
<code>.moveToElementText()</code>	Textbereich in ein Element/Objekt bewegen, das Text enthalten darf per <code>textrange</code> Objekt nur unter Windows 32-Bit
<code>.moveToPoint()</code>	Inhalt des Textbereiches um eine Pixelspanne verschieben (Offset) relativ zur linken oberen Fensterecke nach Verschiebung kann Textbereich leer sein per <code>textrange</code> Objekt nur unter Windows 32-Bit

Beispiel:

```
<SCRIPT LANGUAGE="JScript" FOR=document EVENT=onclick >
    var TextBereich = document.body.createTextRange();
    TextBereich.moveToPoint(window.event.x, window.event.y);
    TextBereich.expand("word");
    TextBereich.select();
</SCRIPT>
```

<code>.namedItem()</code>	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
---------------------------	--

Beispiel:

```
<DIV ID="ID_Div">Dieser Text wird geändert</DIV>
<BUTTON onclick="document.all.namedItem('ID_Div').innerText='Neuer Text!';">
    Klick mich
</BUTTON>
```

<code>.namedRecordset()</code>	Zeiger desjenigen Datenquellen-Record-Objektes mit Namen, das den Standard-Record-Set enthält Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an Eigenschaft <code>.recordset</code> liefert den Zeiger für den Standard-Record-Set in einem Datasource-Objekt (DSO)
--------------------------------	---

Beispiel 1:

```
<SCRIPT>
    // Fired when all the data is available
    function TabelleFuellen()
    {
        var EventObjekt = window.event;

        // prüfen ob nicht Standard-Recordset anliegt (hat keinen Namen)
        if (EventObjekt.qualified != "")
        {
            // ersten Recordset hinter dem Standard-Recordset referenzieren
            var RecordSetObjektMitName=
                EventObjekt.srcElement.namedRecordset(EventObjekt.qualified);

            // auf ersten Satz im Recordset positionieren
            RecordSetObjektMitName.MoveFirst();

            // alle Sätze abklappen im Recordset
            for (int i = 0; i < RecordSetObjektMitName.RecordCount; i++)
            {
                // erstes Feld des Satzes auslesen
                var Wert = RecordSetObjektMitName.Fields(0).value;

                // nächsten Satz im Recordset
                RecordSetObjektMitName.MoveNext();
            }
        }
    }
</SCRIPT>
```




```

<OBJECT CLASSID="clsid:00000000-0000-0000-0000-000000000000"
  ID="ID_Objekt"
  ondatasetcomplete="TabelleFuellen()"
>

<!--Zelle A1:A7 füllen per Recordset -->
<TABLE DATASRC="#ID_Objekt.A1:A7">
  <TR>
    <TD>
      <SPAN DATAFLD="A">
        </SPAN>
      </TD>
    </TR>
  </TABLE>

```

Beispiel 2 für XML:

```

<XML ID="ID_XML">
  <customers>
    <customer>
      <order ID="1">
        <item>
          <name>Butter</name>
          <quantity>12</quantity>
        </item>
        <item>
          <name>Kaese</name>
          <quantity>12</quantity>
        </item>
      </order>
      <order ID="2">
        <item>
          <name>Wurst</name>
          <quantity>100</quantity>
        </item>
      </order>
    </customer>
  </customers>
  <customer>
    <order ID="3">
      <item>
        <name>Kaese</name>
        <quantity>20</quantity>
      </item>
      <item>
        <name>Quark</name>
        <quantity>20</quantity>
      </item>
    </order>
  </customer>
</XML>

<SCRIPT>
  // Pfad zu den Mengendaten ist:
  //      order item name quantity

  function Summieren(NameAlsKette)
  {
    var Summe = 0;

    // Standard-Recordset adressieren get the default data member
    var StandardRecordSet = ID_XML.recordset; // entspricht ID_XML.namedRecordset("");

    // ersten Satz positionieren
    StandardRecordSet.MoveFirst();

    for (var SatzZahler = 0; SatzZahler < StandardRecordSet.RecordCount; SatzZahler++)
    {
      var OrderRecordSet = ID_XML.namedRecordset("", "order");
      OrderRecordSet.MoveFirst(); // ersten Satz
    }
  }

```



```

    for ( var OrderZahler = 0;
        OrderZahler < OrderRecordSet.RecordCount;
        OrderZahler ++
    )
    {
        var ItemRecordSet = ID_XML.namedRecordset("", "order_item");
        ItemRecordSet.MoveFirst(); // ersten Satz

        for ( var ItemZahler = 0;
            ItemZahler < ItemRecordSet.RecordCount;
            ItemZahler ++
        )
        {
            // prüfen ob Funktionsargument NameAlsKette gefunden
            if (ItemRecordSet.Fields("name").value == NameAlsKette)
            {
                // ja also kumulieren aus dem Feld quantity
                Summe +=
                    parseInt(ItemRecordSet.Fields("quantity").value);
            }

            // nächster Satz
            ItemRecordSet.MoveNext();
        }

        // nächster Satz
        OrderRecordSet.MoveNext();
    }

    // nächster Satz
    StandardRecordSet.MoveNext();
}

return Summe;
}

alert( Summieren ("Wurst");)
alert(Summieren ("Kaese");)
</SCRIPT>

```

.navigate() HTTP-Verzeichnis im aktuellen Fenster anzeigen

.navigate() lädt neues HTML-Dokument laut Url in das Fenster
entspricht location.href = "...."
siehe Objekt window

Beispiele: navigate("index.html");
navigate ("file:///c:/index.html");

.NavigateAndFind() Webseite laden, dann Text dort suchen und wenn gefunden, so diesen in der Webseite markieren
(analog zu CTRL-F in der Webseite für Suchen und Finden)
auch Suche in Frames bzw. Unterseiten der Webseite
siehe Objekt window.external

Beispiel

```

<HEAD>
<SCRIPT>
    function Suchen()
    {
        window.external.NavigateAndFind( "http://www.test.de/file.htm",
            ID_Seletct.options[ID_Seletct.selectedIndex].text,
            ""
        );
    }
</SCRIPT>
</HEAD>
<BODY>
    bitte selektieren
    <SELECT ID="ID_Seletct" onchange="Suchen()">
        <OPTION>Eintrag1
        <OPTION>Eintrag2
    </SELECT>
</BODY>

```



<code>.navigateFrame()</code>	HTTP-Verzeichnis im Fenster nach Wahl anzeigen
<code>.navigateHomePage()</code>	Homepage anspringen im Browser
<code>.nextElement()</code>	nächstes Element aus aktiver t:SEQ animieren auf der Timeline wenn kein nächstes Element, so kein Fehler erzeugt siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code>

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:SEQ ID="ID_Seq"
BEGIN="indefinite;"
>

<DIV ID="ID_Div1"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:15px;left:25px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div2"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:30px;left:50px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div3"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:35px;left:75px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div4"
CLASS="time_line_klasse"
DUR="10"
style="position:relative;top:50px;left:100px;width:100px;height:100px;"
>
</DIV>
</t:SEQ>
<BR>
<BUTTON ID="ID_Button1" onclick="ID_Seq.beginElement();"> start</BUTTON>
<BUTTON ID="ID_Button2" onclick="ID_Seq.nextElement();">next </BUTTON>
<BUTTON ID="ID_Button3" onclick="ID_Seq.prevElement();">previous</BUTTON>
<BUTTON ID="ID_Button4" onclick="ID_Seq.endElement();">stop</BUTTON>
</BODY>
</HTML>
```

<code>.nextPage()</code>	nächste Seite des Datasets in Tabelle anzeigen Eigenschaft <code>.dataPageSize</code> muss belegt worden sein siehe Objekt <code>table</code>
--------------------------	---

Beispiel:

Datensätze liegen in der Datei `adress.txt`
Datei `adress.txt` liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815
	Willmann;Theo;1234
	Xantippe;Isa;5678
	Arktin;Richard;1055

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
var SortierRichtung = true;

function Sortieren(FeldBezeichner)
{
```



```

        // Sortierrichtung festlegen
        var Kette = "-"; // Annahme
        if (SortierRichtung) { Kette = "+"; }

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {
                ID_Datenbank.recordset.MoveNext();
            }

            if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
            {
                alert("Letzter Datensatz erreicht!");
            }
        }

        function rueckwaerts(AnzahlSaetze)
        {
            // vorhergehende Seite anzeigen
            document.all.ID_Tabelle.previousPage();

            // und Satzzeiger korrigieren
            for (var i = 0; i < AnzahlSaetze; i++)
            {
                if (ID_Datenbank.recordset.AbsolutePosition > 1)
                {
                    ID_Datenbank.recordset.MovePrevious();
                }

                if (ID_Datenbank.recordset.AbsolutePosition == 1)
                {
                    alert("Erster Datensatz erreicht!");
                }
            }
        }

        // -->
        </SCRIPT>
        </HEAD>
        <BODY>
        <OBJECT ID="ID_Datenbank"
            CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"

        >
            <PARAM NAME="DataURL" VALUE="adress.txt">
            <PARAM NAME="UseHeader" VALUE="True">
            <PARAM NAME="FieldDelim" VALUE=";">
        </OBJECT>

        <TABLE ID="ID_Tabelle"
            DATASRC=#ID_Datenbank
            DATAPAGESIZE=1

        >
            <THEAD>
                <TR>
                    <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
                    <TH><A HREF="JavaScript:Sortieren('name')">Name</A></TH>
                    <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon</A></TH>
                </TR>
            </THEAD>
            <TBODY>
                <TR>
                    <TD><DIV DATAFLD="vorname"></DIV></TD>
                    <TD><DIV DATAFLD="name"></DIV></TD>
                    <TD><DIV DATAFLD="telefon"></DIV></TD>

```



```

        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT          TYPE="button"
                VALUE="Zurueck"
                onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->

<INPUT          TYPE="button"
                VALUE="Vorwaerts"
                onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 //-->
</BODY>
</HTML>

```

`.nextTrack()` nächstes `playItem` Objekt aktivieren laut Behavior-Collection `.style.time2.playList`
 also Abspielen des Tracks starten
 bzw. nächste Wiederholung starten wenn `.repeatCount > 0` und noch nicht
 alle Wiederholungen abgearbeitet wurden
 wenn letzter Track laut der Playliste aktiv ist und dann der nächste Track aktiviert werden soll,
 so wird nicht der erste Track abgespielt, sondern der aktive Track in der
 Wiedergabe gestoppt und danach keine weiteren Aktionen mehr
 Track entspricht `playItem` Objekt
 Trackliste liegt in der Behavior-Collection `.style.time2.playList` als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
 aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
 aktiver Track in der Liste: wird abgespielt
 Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
 siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel 1:

```
object.playList.nextTrack()
```

Beispiel 2:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playList.activeTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playList.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playList.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playList.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playList.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playList.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playList.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playList.activeTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()

```



```

    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA          ID="ID_Media"
                      SRC="test.asx"
                      BEGIN="indefinite"
                      TIMEACTION="visibility"
                      onend="ButtonUpdate();"
                      ontrackchange="AnzeigeUpdate();"
                      onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN   ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN   ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN   ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN   ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN   ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN   ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN   ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN   ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON          ID="ID_Button1"
                    onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON          ID="ID_Button2"
                    onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON          ID="ID_Button3"
                    onclick="ID_Media.playlist.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON          ID="ID_Button4"
                    onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>
</HTML>

```

.normalize() Normalisierung des DOM zur Erreichung einer konsistenten Struktur
 Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen

Beispiel:

```

<HTML>
<BODY onload="ID_Div.normalize();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

Number() konvertiert eine Instanz zu einem numerischen Wert (Number-Objekt-Wert)



Beispiel:

```
var DatumJetzt = new Date();
alert (Number(DatumJetzt));
```

.open()

Dokument instanzieren und mit/ohne Fenster öffnen und Referenz liefern auf Dokument
Ausgabe-Datenstream öffnen und Anzeige des Dokumentes
Instanzieren ohne .open(): siehe .write() und .writeln()
Hinweis: neues Fenster erzeugen als unterstes der aktuellen Fensterhierarchie
und dann Fenster öffnen (anzeigen, rendern)

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    function ErzeugeZurLaufZeit()
    {
        // Dokumentinstanz erzeugen
        var ID_Dokument = document.open("text/html", "replace");

        // Dokumentinhalt festlegen und anzeigen
        ID_Dokument.write("<HTML><BODY>Hallo</BODY></HTML>");

        // Dokumentinstanz schliessen
        ID_Dokument.close();
    }
</SCRIPT>
</HEAD>
<BODY>
<INPUT TYPE="button" onclick=" ErzeugeZurLaufZeit();">
</BODY>
</HTML>
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
    FensterZeiger.close();
}

....

// Fenster öffnen
var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
var FensterDokumentZeiger = FensterZeiger.document;

var Kette= '<HTML>'
+ '<HEAD></HEAD>'
+ '<BODY >'
+ '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
+ '<BR>'
+ '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
+ ' onclick="opener.OnClickHandler();"'
+ '>'
+ '</BODY>'
+ '</HTML>';

FensterDokumentZeiger.open("text/html");
FensterDokumentZeiger.write(Kette);
FensterDokumentZeiger.close();
```

.open()

neues Fenster als unterstes der aktuellen Fensterhierarchie erzeugen
und dann oberstes sichtbare Fenster öffnen (anzeigen, rendern und als aktuell setzen)
nicht möglich bei Dialog-Fenster oder Popup-Fenster
Hinweis: Zeiger des Fensters, das .open() ausführt, liegt in der Eigenschaft .opener
des neu erzeugten Fensters
Daten per '?' + escape() **nicht** übergebbar

siehe Objekt window

Beispiel:

```
window.open("Sample.htm",null, "height=200,width=400,status=yes,toolbar=no,menubar=no,location=no");
```

Beispiel:

```
function OnClickHandler()
{
    alert(FensterZeiger.ID_TextArea.value;
```



```

        FensterZeiger.close();
    }

    ....

    // Fenster öffnen
    var FensterZeiger=window.open("", null, "height=250,width=300,status=no,toolbar=no,menubar=no,location=no");
    var FensterDokumentZeiger = FensterZeiger.document;

    var Kette= '<HTML>'
        + '<HEAD></HEAD>'
        + '<BODY>'
        + '<TEXTAREA ID="ID_TextArea" ROWS="5" COLS="20"></TEXTAREA>'
        + '<BR>'
        + '<INPUT TYPE="button" VALUE="Text an Aufrufer übergeben"'
        + ' onclick="opener.OnClickHandler();"'
        + '>'
        + '</BODY>'
        + '</HTML>';

    FensterDokumentZeiger.open("text/html");
    FensterDokumentZeiger.write(Kette);
    FensterDokumentZeiger.close();

```

.OpenAsTextStream() vorhandene Datei als Text öffnen zum Lesen und Schreiben oder Append, wenn Dateiattribute es zulassen (sonst Fehler erzeugt)
Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```

var DateiNameMitPfad = "c:\\test.txt";
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var Datei = DateiSystem.GetFile(DateiNameMitPfad);
var DateiOffen = Datei.OpenAsTextStream(2,0);
DateiOffen.Write( "neuer Text" );
DateiOffen.Close();
DateiOffen = Datei.OpenAsTextStream(1,2);
var Kette = DateiOffen.ReadLine( );
DateiOffen.Close();
alert(Kette);

```

.OpenTextFile() Datei als Text öffnen zum Lesen, Schreiben oder Append (Schreiben durch Anhängen)
Schreiben und Schliessen der Datei per Objekt FileSystemObject.TextStream

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0)
DateiOffen.WriteLine("Test");
DateiOffen.Close();

```

.parentElement() Zeiger auf das Elternelement (Container) des Textbereiches liefern
Hinweis bei Elementverschachtelung:
es wird das direkt um den Textbereich liegende Element referenziert
per textrange Objekt
nur unter Windows 32-Bit

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
    var SelectionObjekt = document.selection; // Eltern
    var TextBereichDerSelection = SelectionObjekt.createRange();
    var ZeigerAufSelection = TextBereichDerSelection.parentElement();
    alert(ZeigerAufSelection.tagName);
</SCRIPT>

```

.parentTimeToActiveTime() Wert der Timeline der Eltern zum korrespondierenden Wert der aktives Timeline des Elementes konvertieren
per Eigenschaft .isActive das Element auf Aktivsein prüfen
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL REPEATCOUNT="3"

```




```

>
    <DIV ID="ID_Div"
        CLASS="time_line_klasse"
        BEGIN="1s"
        DUR="5s"
    >
    </DIV>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button"
    onclick=" alert( 'Punkt auf der aktiven Timeline: ' + ID_Div.parentTimeToActiveTime(3) ); "
>
    parentTimeToActiveTime(3);
</BUTTON>
</BODY>
</HTML>

```

.parentTimeToDocumentTime() Wert der Timeline der Eltern des Elementes in den korrespondierenden Wert der Timeline des Dokumentes konvertieren
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:PAR ID="ID_Par"
        CLASS="time_line_klasse"
        REPEATCOUNT="3"
    >
        <DIV ID="ID_Div"
            CLASS="time_line_klasse"
            BEGIN="2s"
            DUR="2s"
        >
        </DIV>
    </t:PAR>
    <SPAN ID="ID_Span">
        parentTimeToDocumentTime:
    </SPAN>
    <BR>
    <BUTTON ID="ID_Button"
        onclick=" ID_Span.innerText='parentTimeToDocumentTime: '
            + ID_Div.parentTimeToDocumentTime(3);"
    >
        parentTimeToDocumentTime(3)
    </BUTTON>
</BODY>
</HTML>

```

.parse() Datum als String parsen und Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit liefern
Datum als String in den Basis-Datentyp date konvertieren
Script-Objekt Date

Beispiele:

```

"Jan 5, 1996 08:47:00"
"November 1, 1997 10:15:00 "
"November 1, 1997 10:15 AM"
"7/20/96 08:47:00"
"7-20-96 08:47:00"
"Tuesday November 9 1990"
"7-20-96 08:" es muss Doppelpunkt kodiert sein, damit 08 als Stundenangabe interpretiert wird
"7-20-96 08:47 GMT"

```

parseFloat() String oder Literal parsen und nach Floating-point umwandeln
können enthalten
Vorzeichen + und -
Ziffern 0 bis 9
Dezimalkomma als Punkt
e oder E
Blanks (werden ignoriert)



falls andere Zeichen enthalten sind, gilt:

String bzw. Literal bis vor die erste fehlerhaften
Stelle geparkt und dann konvertiert

Erfolg der Konvertierung ist per Methode isNaN() zu prüfen

Beispiele:

gültiges Literal

```
parseFloat("3.14")
parseFloat("314e-2")
parseFloat("0.0314E+2")
```

gültiger String

```
var x = "3.14"
parseFloat(x)
```

ungültiges Literal

```
parseFloat("FF2")
```

```
alert(parseFloat("17.50 DM"));
```

Beispiel für Ziffern-Zeichenkettenwert nach numerisch und dabei Dezimalkomma zu Dezimalpunkt umwandeln:

Es wird angenommen, dass genau ein Komma vorkommt.

```
function punkt_zu_komma(zeichenkette)
{
    var pos_komma = zeichenkette.indexOf(",");
    var funktionswert;

    if(pos_komma == -1)
    {
        if (zeichenkette.indexOf(".") == -1)
        { funktionswert = parseInt(zeichenkette); }
        else
        { funktionswert = parseFloat(zeichenkette); }
    }
    else
    {
        funktionswert = parseFloat(
            zeichenkette.substring(0, pos_komma)
            + "."
            + zeichenkette.substring(pos_komma + 1, zeichenkette.length)
        );
    }

    return funktionswert;
}
```

parseInt()

String oder Literal

parsen und nach Integer umwandeln
können enthalten

Vorzeichen + und -
Ziffern 0 bis 9, aber keine Vornull(en)
Blanks (werden ignoriert)
eventuelle Buchstaben A bis F

falls andere Zeichen enthalten sind, gilt:

String bzw. Literal bis vor die erste fehlerhaften Stelle geparkt und dann
konvertiert

Erfolg der Konvertierung ist per Methode isNaN() zu prüfen

Beispiel:

gültige Literale:

```
parseInt("F", 16)
parseInt("17", 8)
parseInt("15", 10)
parseInt(15.99, 10)
parseInt("FXX123", 16)
parseInt("1111", 2)
parseInt("15*3", 10)
parseInt("17")
parseInt("0x7", 16)
parseInt("0x7")
parseInt("0")
parseInt("0x11", 16)
parseInt("0x11", 0)
parseInt("0x11")
```



ungültige Literale:

```
parseInt("F")
parseInt("Hello", 8)
parseInt("0x7", 10)
parseInt("FFF", 10)
parseInt('002')
```

.pasteHTML()

Textbereich-Inhalt durch Text ersetzen oder leeren Textbereich füllen
Text kann auch HTML enthalten
Tabelle nur mit kompletten HTML-Code einfügbar, also z.B. keine einzelne Zelle
Achtung: Wenn HTML-Code im Text enthalten ist, muss das Elternobjekt
auch diesen ansich unterstützen
Bsp.: textArea erlaubt kein HTML-Code
HTML-Code wird geparkt
per textrange Objekt
nur unter Windows 32-Bit

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
// Selektion erzeugen
var SelectionObjekt = document.selection;

// prüfen ob erzeugt wurde
if (SelectionObjekt!=null)
{
    // wurde erzeugt
    var TextBereichDerSelektion rng = SelectionObjekt.createRange();

    // prüfen ob erzeugt wurde
    if (TextBereichDerSelektion!=null)
    {
        // wurde erzeugt, also Selektion füllen
        TextBereichDerSelektion.pasteHTML("<P><B>Text der Selektion</B></P>");
    }
}
</SCRIPT>
```

.pause()

deprecated
auf Timeline pausieren

.pauseElement()

aktives Element auf Timeline pausieren lassen
ersetzt Methode pause(), die deprecated ist und nicht mehr verwendet werden darf !
per Eigenschaft .isActive das Element auf Aktivsein prüfen
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
.TextAufHighLight{ color:#CCCCC; font-weight:bold;}
</STYLE>
<SCRIPT>
function PauseAufheben()
{
    ID_Table.resumeElement();
    ID_Button1.disabled = false;
    ID_Button2.disabled = true;
}

function Pausieren()
{
    ID_Table.pauseElement();
    ID_Button1.disabled = true;
    ID_Button2.disabled = false;
}
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<TABLE ID="ID_Table"
BORDER="1"
CLASS="time_line_klasse"
DUR="8s"
REPEATCOUNT="indefinite"
```



```

        TIMECONTAINER="SEQ"
        TIMEACTION="none"
    >
    <TBODY>
        <TR>
            <TH WIDTH="120">Eins</TH>
            <TH WIDTH="50">Zwei</TH>
            <TH WIDTH="40">Drei</TH>
        </TR>
        <TR
            ID="ID_TR1"
            CLASS="time_line_klasse"
            TIMEACTION="class: TextAufHighLight "
            DUR="2s"
        >
            <TD>EinsA</TD>
            <TD>ZweiA</TD>
            <TD>DreiA</TD>
        </TR>
        <TR
            ID="ID_TR1"
            CLASS="time_line_klasse"
            TIMEACTION="class: TextAufHighLight "
            DUR="2s"
        >
            <TD>EinsB</TD>
            <TD>ZweiB</TD>
            <TD>DreiB</TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<BUTTON
    ID="ID_Button1"
    onclick="Pausieren();"
>
    pausieren
</BUTTON>
<BUTTON
    ID="ID_Button2"
    DISABLED="true"
    onclick="PauseAufheben();"
>
    Pause aufheben
</BUTTON>
</BODY>
</HTML>

```

.playNext()

Wiedergabe des nächsten Eintrages in der Playliste (Objekt PlaylistInfo) starten
verändert Eigenschaft .playState
Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen.

aktuelle

Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die
Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei.
Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt.
Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende

Methoden nutzbar:
.getItemInfo()
.getAttributeName()
Eigenschaften nutzbar:
.attributeCount

Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.
siehe Methoden .playURL() .stop() und Eigenschaft .playState
siehe Behavior .style.mediaBar

Beispiel:

```

<DIV
    ID="ID_Div"
    STYLE="behavior:url(#default#mediaBar)"
    onopenstatechange="alert(ID_Div.openState);"
>
</DIV>
<INPUT
    TYPE=button
    ID="ID_Input"
    VALUE='abspielen des nächsten Eintrages in der Playliste'
    onclick= "ID_Div.playNext();
        ID_Div.disabledUI = true;
    "
>

```



<code>.playURL()</code>	<p>Laden einer Media-Datei in die Media Bar und wenn Laden erfolgreich, so Wiedergabe der Media-Daten</p> <p>Achtung: Wird die Media-Datei nicht gefunden oder ist der Mime-Typ der Media-Datei unbekannt, dann wird eine Standard-Seite in das Medien-Fenster geladen.</p> <p>verändert Eigenschaft <code>.playState</code></p> <p>Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen.</p> <p>Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die</p>
aktuelle	<p>Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei.</p> <p>Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt.</p> <p>Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:</p> <ul style="list-style-type: none"> <code>.getItemInfo()</code> <code>.getAttributeName()</code> <p>Eigenschaften nutzbar:</p> <ul style="list-style-type: none"> <code>.attributeCount</code> <p>Es ist also vorher die Eigenschaft <code>.playState</code> auf Werte > 10 zu prüfen.</p> <p>siehe Methoden <code>.playNext()</code> <code>.stop()</code> und Eigenschaft <code>.playState</code></p> <p>siehe Behavior <code>.style.mediaBar</code></p>
Beispiel:	<pre> <DIV ID="ID_Div" STYLE="behavior:url(#default#mediaBar)" onopenstatechange="alert(ID_Div.openState);"> > </DIV> <INPUT TYPE=button ID="ID_Input" VALUE='abspielen von test.aspx' onclick= "ID_Div.playURL('http://www.test.de/test.aspx','video/x-ms-asf'); ID_Div.disabledUI = true; " > </pre>
<code>.pop()</code>	<p>letztes Feldelement liefern und danach aus dem Feld entfernen</p> <p>Feld hat die Objektklasse JScript-Objekt Array bzw. Script-Objekt Array</p> <p>ab IE 5.5 und NS 6.x</p> <p>siehe auch <code>.shift()</code></p>
Beispiel:	<pre> var Feld = new Array(0,1,2,3,4); alert (Feld.pop()); // "4" alert(Feld.join()); // "0,1,2,3" </pre>
<code>.pow()</code>	<p>liefert die Potenz von Basis hoch Exponent</p> <p>siehe Script-Objekt Math</p>
<code>.prevElement()</code>	<p>vorhergehendes Element aus aktiver t:SEQ animieren auf der Timeline</p> <p>wenn kein vorhergehendes Element, so wird Timeline von t:SEQ auf 0 gesetzt</p> <p>siehe Objekt <code>currTimeState</code> und Behavior <code>.style.time2</code></p>
Beispiel:	<pre> <HTML XMLNS:t="urn:schemas-microsoft-com:time"> <HEAD> <?IMPORT namespace="t" implementation="#default#time2"> <STYLE> .time_line_klasse { behavior: url(#default#time2) } </STYLE> </HEAD> <BODY> <t:SEQ ID="ID_Seq" BEGIN="indefinite;" > <DIV ID="ID_Div1" CLASS="time_line_klasse" DUR="10" style="position:relative;top:15px;left:25px;width:100px;height:100px;" > </DIV> <DIV ID="ID_Div2" CLASS="time_line_klasse" DUR="10" style="position:relative;top:30px;left:50px;width:100px;height:100px;" > </pre>



```

</DIV>
<DIV ID="ID_Div3"
      CLASS="time_line_klasse"
      DUR="10"
      style="position:relative;top:35px;left:75px;width:100px;height:100px;"
>
</DIV>
<DIV ID="ID_Div4"
      CLASS="time_line_klasse"
      DUR="10"
      style="position:relative;top:50px;left:100px;width:100px;height:100px;"
>
</DIV>
</t:SEQ>
<BR>
<BUTTON ID="ID_Button1" onclick="ID_Seq.beginElement();"> start</BUTTON>
<BUTTON ID="ID_Button2" onclick="ID_Seq.nextElement();">next</BUTTON>
<BUTTON ID="ID_Button3" onclick="ID_Seq.prevElement();">previous</BUTTON>
<BUTTON ID="ID_Button4" onclick="ID_Seq.endElement();">stop</BUTTON>
</BODY>
</HTML>

```

.previousPage() vorhergehende Seite des Datasets in Tabelle anzeigen
 Eigenschaft .dataPageSize muss belegt worden sein
 siehe Objekt table

Beispiel:

Datensätze liegen in der Datei adress.txt
 Datei adress.txt liegt im gleichen Verzeichnis wie nachfolgendes HTML-Dokument
 hat folgenden Inhalt:

1. Satz	vorname;name;telefon
ab 2. Satz	Guericke;"Otto, von";0815 Willmann;Theo;1234 Xantippe;Isa;5678 Arkin;Richard;1055

```

<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JScript">
<!--
    var SortierRichtung = true;

    function Sortieren(FeldBezeichner)
    {
        // Sortierrichtung festlegen
        var Kette = "-";      // Annahme
        if (SortierRichtung) {Kette = "+";}

        // nächste Sortierung umgekehrt
        SortierRichtung = !SortierRichtung;

        // sortieren
        ID_Datenbank.Sort= Kette + FeldBezeichner;

        // und das Ergebnis anzeigen
        ID_Datenbank.Reset();
    }

    function vorwaerts(AnzahlSaetze)
    {
        // nächste Seite anzeigen
        document.all.ID_Tabelle.nextPage();

        // und Satzzeiger korrigieren
        for (var i = 0; i < AnzahlSaetze; i++)
        {
            if (ID_Datenbank.recordset.AbsolutePosition !=
                ID_Datenbank.recordset.RecordCount)
            {ID_Datenbank.recordset.MoveNext();}
        }

        if (ID_Datenbank.recordset.AbsolutePosition == ID_Datenbank.recordset.RecordCount)
        {alert("Letzter Datensatz erreicht!");}
    }

```



```

function rueckwaerts(AnzahlSaetze)
{
    // vorhergehende Seite anzeigen
    document.all.ID_Tabelle.previousPage();

    // und Satzzeiger korrigieren
    for (var i = 0; i < AnzahlSaetze; i++)
    {
        if (ID_Datenbank.recordset.AbsolutePosition > 1)
        {
            ID_Datenbank.recordset.MovePrevious();
        }
    }

    if (ID_Datenbank.recordset.AbsolutePosition == 1)
    {
        alert("Erster Datensatz erreicht!");
    }
}

// -->
</SCRIPT>
</HEAD>
<BODY>
<OBJECT ID="ID_Datenbank"
    CLASSID="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
>
    <PARAM NAME="DataURL" VALUE="adress.txt">
    <PARAM NAME="UseHeader" VALUE="True">
    <PARAM NAME="FieldDelim" VALUE=";">
</OBJECT>

<TABLE ID="ID_Tabelle"
    DATASRC=#ID_Datenbank
    DATAPAGESIZE=1
>
    <THEAD>
        <TR>
            <TH><A HREF="JavaScript:Sortieren('vorname')">Vorname </A></TH>
            <TH><A HREF="JavaScript:Sortieren('name')">Name </A></TH>
            <TH><A HREF="JavaScript:Sortieren('telefon')">Telefon </A></TH>
        </TR>
    </THEAD>
    <TBODY>
        <TR>
            <TD><DIV DATAFLD="vorname"></DIV></TD>
            <TD><DIV DATAFLD="name"></DIV></TD>
            <TD><DIV DATAFLD="telefon"></DIV></TD>
        </TR>
    </TBODY>
</TABLE>
<BR>
<INPUT TYPE="button"
    VALUE="Zurueck"
    onclick=rueckwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->

<INPUT TYPE="button"
    VALUE="Vorwaerts"
    onclick=vorwaerts(1)"
>
<!-- Anzahl der Sätze laut TABLE DATAPAGESIZE=1 -->
</BODY>
</HTML>

```

.prevTrack() vorhergehendes playItem Objekt aktivieren laut Behavior-Collection .style.time2.playList
 also Abspielen des Tracks starten
 aber **nicht** vorhergehende Wiederholung starten wenn .repeatCount > 0 und noch nicht
 alle Wiederholungen abgearbeitet wurden
 wenn erster Track laut der Playliste aktiv ist und dann der vorhergehende Track aktiviert werden soll,
 so wird der erste Track nochmal abgespielt
 Track entspricht playItem Objekt
 Trackliste liegt in der Behavior-Collection .style.time2.playList als Playliste:
 Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline
 aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
 Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
 aktiver Track in der Liste: wird abgespielt
 Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel 1:



```
object.playlist.prevTrack()
```

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
    {
        ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
    }

    function AnzeigeLoeschen()
    {
        ID_Span1.innerText = "Titel: ";
        ID_Span2.innerText = "Autor: ";
        ID_Span3.innerText = "Abstract: ";
        ID_Span4.innerText = "Copyright: ";
        ID_Span5.innerText = "Filename: ";
        ID_Span6.innerText = "Banner: ";
        ID_Span7.innerText = "Banner Abstract: ";
        ID_Span8.innerText = "Banner MoreInfo: ";
    }
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"
        TIMEACTION="visibility"
        onend="ButtonUpdate();"
        ontrackchange="AnzeigeUpdate();"
        onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
```




```

<SPAN ID="ID_Span5">Filename:</SPAN>
<BR>
<SPAN ID="ID_Span6">Banner:</SPAN>
<BR>
<SPAN ID="ID_Span7">Banner Abstract:</SPAN>
<BR>
<SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

<BR>
<BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
>
    Start
</BUTTON>
<BUTTON ID="ID_Button2"
        onclick="ID_Media.playList.nextTrack();"
>
    naechster Track
</BUTTON>
<BUTTON ID="ID_Button3"
        onclick="ID_Media.playList.prevTrack();"
>
    vorhergehender Track
</BUTTON>
<BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"
>
    Stop
</BUTTON>
</BODY>
</HTML>

```

.print() ruft Dialog-Box-Druckfenster auf zum Druck des Dokumentes im aktuellen Fenster
entspricht Druckbutton oder Datei-Menü-Drucken
löst folgende Ereignisse aus: onbeforeprint und onafterprint
siehe Objekt window

Beispiel für Verwendung der Ereignisse per Handler

onbeforeprint Handler blendet Teile der Seite ein/aus, die gedruckt werden sollen
onafterprint Handler hebt Veränderungen von onbeforeprint wieder auf

Beispiel:

```
<INPUT TYPE="button" VALUE="Drucken"onclick="javascript:self.print()">
```

.prompt() Eingabefenster erzeugen:

1. Meldungstext anzeigen,
Eingabezeile vorbelegen und anzeigen,
OK- bzw. CANCEL-Button anzeigen
2. auf User.Eingabe in die Zeile und anschliessendem Druck auf OK warten
siehe Objekt window

.push() optionales Anhängen von Werten an das Ende des Feldes (auch wenn das Feld leer ist)
Anzahl der Feld-Elemente liefern (auch nach optionalem Anhängen)
Feld hat die Objektklasse JScript-Objekt Array bzw. Script-Objekt Array
ab IE 5.5 und NS 6.x
siehe auch .concat()

Beispiel:

```

var Feld1            = new Array(0,1);
var Feld2_Quelle     = new Array(2,3);
var Wert_Quelle      = 4;
alert(Feld1.push(Wert_Quelle , Feld2_Quelle));      // 4 und nicht 5 !
alert(Feld1.join());    // "0,1,2,3"                    // 4 wird nicht angehangen

```

.queryCommandEnabled() prüfen ob Kommando ausführbar ist

.queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht

.queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht

.queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird

.queryCommandValue() Wert eines Kommandos liefern

.random() Zufallszahl liefern



0<= zufallszahl <1
siehe Script-Objekt Math

.Read() in der offenen Datei die nächsten Zeichen lesen (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.Read(6));
DateiOffen.Close();
```

.ReadAll() offene Datei komplett auslesen (auf einen Schlag)
nur direkt nach Dateiöffnen möglich

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.ReadAll());
DateiOffen.Close();
```

.ReadLine() in der offene Datei die nächste Zeile lesen (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
alert(DateiOffen.ReadLine());
DateiOffen.Close();
```

.recalc() dynamischen Eigenschaften des Dokumentes neu berechnen
Hinweis: andere Objekte, die Eigenschaften des Dokumentes nutzen, werden auch neu berechnet,
wenn Eigenschaften nicht in einem Berechnungsausdruck vorliegen

Beispiel 1:

```
<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var seconds = 0;

    function init()
    {
        ID_Div1.style.setExpression("width","seconds*10");
        ID_Div2.setExpression("innerText","seconds.toString()");
    }

    function timer()
    {
        seconds++;
        document.recalc();
    }

    function starttimer()
    {
        if (timerID == null)
        {
            timerID = setInterval("timer()", 1000);
            ID_Button1.disabled = true;
            ID_Button2.disabled = false;
        }
    }

    function stoptimer()
    {
        if (timerID != null)
```



```

        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

    function resettimer()
    {seconds = 0;}
</SCRIPT>
</HEAD>
<BODY onload="init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div2" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"                onclick="starttimer()">Start Timer</BUTTON><BR>
    <BUTTON ID="ID_Button2" DISABLED="true" onclick="stoptimer()">Stop Timer</BUTTON><BR>
    <BUTTON                                onclick="resettimer()">Reset Timer</BUTTON><BR>
</BODY>
</HTML>

```

Beispiel 2 für Sekundenbalken:

```

<HTML>
<HEAD>
<STYLE>
    BUTTON { font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText","Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler ++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
        }
    }

```



```

        ID_Button1.disabled = false;
        ID_Button2.disabled = true;
    }

    function ZurueckSetzen()
    { Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"

    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"

    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"

    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

Beispiel 3 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln
    // Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    // ++++++ Routinen der Sekundenzählung
    var SekundenZahler = 0;
    var SekundenZahlerTimeoutID = null;

    function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
    {
        // Zähler erhöhen
        SekundenZahler++;

        // visuelle Anzeige im Dokument auffrischen, also alle Ausdrücke der Style-Werte
        // neu berechnen und damit alle DIV's neu visualisieren
        document.recalc();
    }

    function SekundenZaehlen_Start()
    {
        // prüfen ob Sekundenzählen nicht bereits läuft
        if (SekundenZahlerTimeoutID == null)
        {
            // nicht aktiv

```



```

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekunden zählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString()+ " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init

```



```

        SekundenZahler          = 0;
        SekundenZahlerTimeoutID = null;

        // ---- visuelle Anzeige erzeugen
        // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
        //       Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
        //       Der Ausdruck liefert den Wert , welcher sofort das Layout der
        //       DIV's beeinflusst.
        //       Jeder Ausdruck besitzt den SekundenZahler als Komponente.
        //       Damit ändert sich der Wert des Ausdrucks.
        //       Für die Neuberechnung des Ausdrucks ist der Aufruf von
        //       document.recal()
        //       nötig.
        //       Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
        //       Damit wird der Style-Wert permanent neu berechnet.
        //       Damit visualisieren sich die DIV's permanent neu.

        // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
        //       also dynamisch anzeigen
        ID_DIV_Balken.style.setExpression( "width",
                                           "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                           );

        // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
        //       also dynamisch anzeigen
        ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

        // - - - Messlatte statisch anzeigen
        ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
        ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                                       + SoundDauerInSekunden.toString()
                                       + " Sekunden";
    }

    // ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
    if (ie)
    {
        document.write('<BODY></BODY>');

        document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

        document.write( ' <DIV ID="ID_DIV_Balken"'
                        + ' STYLE="background-color:lightblue"'
                        + '>'
                        + '</DIV>'
                        + '<BR>'

                        );

        document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
                        + ' STYLE="color:hotpink;font-weight:bold"'
                        + '>'
                        + '</DIV>'
                        + '<BR>'

                        );

        document.write( ' <DIV ID="ID_DIV_MessLatte"'
                        + ' STYLE="color:white;background-color:gray"'
                        + '>'
                        + '</DIV>'

                        );

        // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

        // ---- Sound-Objekt erzeugen anhand globaler Variablen
        SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

        // ---- Sound-Objekt wiedergeben
        SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
    }
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben!-->
</BODY>

```



</HTML>

.refresh() Anzeige der Tabelle neu erzeugen
 Änderungen an der Tabelle sichtbar machen z.B. wenn Tabelle in Anzahl Zeilen bzw. Spalten manipuliert wurde
 siehe Objekt table

.releaseCapture() Maus-Überwachung ausschalten für ein Objekt
 Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
 Hinweis: einschalten per Methode .setCapture()

Beispiel:

```
<BODY onload="ID_Div.setCapture();"
onclick="document.releaseCapture();"
>
<DIV ID="ID_Div"
onmousemove="ID_Textarea.value = event.clientX + event.clientY;"
onlosecapture="alert(event.srcElement.id + ' hat keine Mausüberwachung mehr');"
>
<TEXTAREA ID="ID_Textarea" COLS=2>Test</TEXTAREA>
</DIV>
</BODY>
```

.reload() aktuelles Dokument neu laden
 false Default
 Dokument aus dem Browsercache laden
 true Dokument vom Server laden
 Hinweis: Server kann eigenen Cache haben

Beispiel:

```
<HEAD>
<SCRIPT>
function Entfernen()
{
    // versuche den Text zu entfernen
    try
    {
        var KindZeigerAufTextImDiv = ID_Div.children(0);
        ID_Div.removeChild(KindZeigerAufTextImDiv);
        // Achtung: Der Text ist noch sichtbar !!!!
    }
    // oder fange das Ereignis des bereits entfernten Textes ein
    // und behandle das Ereignis
    catch(x)
    {
        alert( "Text wurde entfernt !\n"
            + "Das Dokument muss neu geladen werden !
        );
        document.location.reload();
    }
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div" onclick="Entfernen()">
    Klick, um diesen Text zu entfernen !
</DIV>
</BODY>
```

.remove() ein Element aus einer Collection entfernen

.Remove() genau eine Date aus dem Dictionary Objekt entfernen (Schlüssel- und Datenfeld entfernen)

Beispiel:

```
var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchlüsselVorhanden(DatenSchlüssel)
{return DatenSpeicher.Exists(DatenSchlüssel);}

// Daten-Elemente erzeugen
var DatenSchlüssel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchlüsselVorhanden(DatenSchlüssel)))
{
```



```

// Schlüssel nicht vorhanden, also Date hinzufügen
DatenSpeicher.add (DatenSchluessel, Date);

// und Date anzeigen
alert(DatenSpeicher.Item(DatenSchluessel));

// und löschen
DatenSpeicher.remove(DatenSchluessel);

// und Meldung
alert(SchluesselVorhanden(DatenSchluessel));
}

```

.RemoveAll() alle Daten aus dem Dictionary entfernen (alle Schlüssel- und Datenfelder), so dass das Dictionary Objekt leer aber weiterhin instanziiert ist

Beispiel:

```

var DatenSpeicher = new ActiveXObject("Scripting.Dictionary");

function SchluesselVorhanden(DatenSchluessel)
{return DatenSpeicher.Exists(DatenSchluessel);}

// Daten-Elemente erzeugen
var DatenSchluessel = "a";
var Date = "test"

// prüfen ob Schlüssel noch nicht vorhanden ist
if ( ! (SchluesselVorhanden(DatenSchluessel)))
{
    // Schlüssel nicht vorhanden, also Date hinzufügen
    DatenSpeicher.add (DatenSchluessel, Date);

    // und Date anzeigen
    alert(DatenSpeicher.Item(DatenSchluessel));

    // und löschen
    DatenSpeicher.removeAll();

    // und Meldung
    alert(SchluesselVorhanden(DatenSchluessel));
}

```

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode `.createAttribute()` erzeugte Attribute werden nicht erfasst
DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Entfernen()
    {
        ID_Div.removeAttribute("TITLE");
    }
</SCRIPT>
</HEAD>
<BODY onload="Entfernen();">
    <DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>

```

.removeAttributeNode() entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern
DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function ToolTipKnotenEntfernen()
    {
        var Knoten = ID_Div.getAttributeNode("TITLE");
        ID_Div.removeAttributeNode(Knoten);
    }
</SCRIPT>
</HEAD>

```




```
<BODY onload="ToolTipKnotenEntfernen();">
  <DIV ID="ID_Div" TITLE="Tooltip-Text ">Es ist ein Tooltip-Text vorhanden</DIV>
</BODY>
</HTML>
```

`.removeBehavior()` per Methode `.addBehavior()` einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
DOM wird geändert

Beispiel:

```
<SCRIPT>
var FeldDerEigenschaftenID      = new Array(); // für removeBehavior
var FeldDerTagsLimDokument      = new Array ();
var FeldDerTagsLimDokument_Laenge = 0;

function EigenschaftHinzufuegen()
{
    FeldDerTagsLimDokument      = document.all.tags ("LI");
    FeldDerTagsLimDokument_Laenge = FeldDerTagsLimDokument.length;
    for (i=0; i < FeldDerTagsLimDokument_Laenge; i++)
    {
        var EigenschaftenID // immer neu anlegen wegen Zeigerprüfung
        = FeldDerTagsLimDokument [i].addBehavior ("hilite.htc");

        if (iEigenschaftenID)
        {FeldDerEigenschaftenID[i] = EigenschaftenID;}
    }
}

function EigenschaftEntfernen()
{
    for (i=0; i < FeldDerTagsLimDokument_Laenge; i++)
    {FeldDerEigenschaftenID[i].removeBehavior (FeldDerEigenschaftenID[i]); }
}
</SCRIPT>
<A HREF="javascript:EigenschaftHinzufuegen()">Eigenschaft hinzufuegen</A>
<A HREF="javascript:EigenschaftEntfernen()">Eigenschaft entfernen</A>
```

`.removeChild()` Kind-Objekt aus einem Objekt entfernen aus DOM und Referenz auf das entfernte Kind liefern
Sichtbarkeit erst, wenn Ende-Tag geparkt wurde, also das Dokument neu geladen wurde
DOM wird geändert

Beispiel:

```
<HEAD>
<SCRIPT>
function Entfernen()
{
    // versuche den Text zu entfernen
    try
    {
        var KindZeigerAufTextImDiv = ID_Div.children(0);
        ID_Div.removeChild(KindZeigerAufTextImDiv);
        // Achtung: Der Text ist noch sichtbar !!!!
    }
    // oder fange das Ereignis des bereits entfernten Textes ein
    // und behandle das Ereignis
    catch(x)
    {
        alert(      "Text wurde entfernt !\n"
        + "Das Dokument muss neu geladen werden !
        );
        document.location.reload();
    }
}
</SCRIPT>
</HEAD>
<BODY>
  <DIV ID="ID_Div" onclick=" Entfernen()">
    Klick, um diesen Text zu entfernen !
  </DIV>
</BODY>
```

`.removeExpression()` Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form
objekt.style.eigenschaft.
dient.



Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
DOM wird nicht geändert

Beispiel: aus Platzgründen die Zeichenkette von STYLE umgebrochen,
was eigentlich nicht zulässig ist, und es sind nicht alle SPAN kodiert.

```

ID_Span.style.setExpression("width","document.body.style.fontSize");
<SPAN ID="ID_Span"
  STYLE="background-color:lightgreen;
        width:expression( trueBlueSpan.style.pixelWidth
                          + oldYellowSpan.style.pixelWidth
                          )
">
</SPAN>
ID_Span.style.removeExpression("width");

```

.removeNamedItem() Attribut entfernen anhand Attributname (analog zu ID oder NAME-Attribut),
wobei danach der Standard-Attributwert automatisch weiterverwendet wird
(falls Standard vorhanden ist),
und Zeiger auf gelöscht Attribut liefern
ab IE 6

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
  function Entfernen()
  {
    var ZeigerAufFeld = ID_Div.attributes;
    ZeigerAufFeld.removeNamedItem("TITLE");
  }
</SCRIPT>
</HEAD>
<BODY>
  <DIV onclick="Entfernen();" ID="ID_Div" TITLE="Tooltip-Text ">
    Klick um den Tooltip-Text zu entfernen
  </DIV>
</BODY>
</HTML>

```

.removeNode() Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern
Sichtbarkeit erst wenn Ende-Tag geparkt wurde
DOM wird geändert

Beispiel:

```

<SCRIPT>
  function Entfernen()
  {Tabelle.removeNode(true);}
</SCRIPT>
<TABLE ID = "Tabelle" >
<TR>
  <TD>Zelle 1</TD>
  <TD>Zelle 2</TD>
</TR>
</TABLE>
<INPUT TYPE = button VALUE = " Entfernen" onclick = " Entfernen()">

```

.removeRule() StyleSheet aus der Collection styleSheet.rules entfernen
Achtung: Um Style auch sichtbar zu entfernen, muss
Dokument muss neu geladen werden
oder alle betroffene Style-Elemente neu mit Wert belegen
durch Zuweisung auf sich selbst (siehe Beispiel)
siehe Objekt styleSheet und Collection styleSheet.rules

Beispiel:

```

<STYLE>
  P {color:green}
</STYLE>
<SCRIPT>
  function StyleEntfernen()
  {
    var StyleSheetsObjekt = document.styleSheets;
    var AnzahlStyleSheets = StyleSheetsObjekt.length;

    if (AnzahlStyleSheets > 0)
    {

```



```

// StyleSheet mit Index 0 bearbeiten also P {color:green}
var StyleSheetAnIndex0 = StyleSheetsObjekt[0];

// wobei P {color:green} eine Regel ist, also Collection rules verwenden
var StyleSheetsRegelCollection = StyleSheetAnIndex0.rules;

var AnzahlRegeln = StyleSheetsRegelCollection.length;

if (AnzahlRegeln > 0)
{
    // Regel P {color:green} entfernen
    StyleSheetAnIndex0.removeRule(0);

    // visuell auch die Farbe entfernen durch Zuweisung auf sich selbst also
    // nun ohne die Regel P {color:green}
    ID_P.innerHTML= ID_P.innerHTML;
}
}
</SCRIPT>

<P ID="ID_P" >Test</P>
<BUTTON onclick="StyleEntfernen()">Text im P-Tag entfaerben.</BUTTON>

```

`.replace()` neues Dokument zuweisen und laden
im Verlauf (History) wird der Eintrag des alten, vorherigen Dokumentes durch
den Eintrag des neuen zu ladenden Dokumentes ersetzt.
Altes Dokument isrt damit **nicht** mehr per Vorwärts- und Zurück-Button einstellbar.

Beispiel:

```
window.location.replace("test.html");
```

`.replace()` Suche in einem String oder String-literal per RegExp Objekt (siehe dort Methode `.exec()`)
und gefundenen String ersetzen
siehe Script-Objekt String

Beispiel 1:

```

var Kette =
    "The man hit the ball with the bat.\nwhile the fielder caught the ball with the glove.";
var RegExpressionAusdruck = /The/g;
var Feld1 = Kette.replace(RegExpressionAusdruck, "A"); // "A" ersetzt "The"
var Feld2 =
    "The man hit the ball with the bat.while the fielder caught the ball with the glove.".replace(
        RegExpressionAusdruck, "A"); // "A" ersetzt "The"

```

Beispiel 2:

```

function F_ahrenheitTauschenGegen_C_eslius(Kette)
{
    var RegExpressionAusdruck = /(d+(\.d*)?)F\b/g;

    return ( Kette.replace(
        RegExpressionAusdruck,
        function($0,$1,$2) {return(((($1-32) * 5/9) + "C");}
    )
    );
}
document.write(F_ahrenheitTauschenGegen_C_eslius ("Wasser kristallisiert bei 32F und siedet bei 212F."));

```

Beispiel 3:

```

var zu_durchsuchende_kette = "Otto Waalkes"; // oder RegExp.input="Otto Waalkes"
var such_muster = /(wOtto)/g; // such_muster ist ein regulärer Ausdruck
// (Objekt RegExp)
// Suchmuster ist Otto, wobei Otto gefunden
// werden muss für eine erfolgreiche Suche
// anstelle von " ist / zu kodieren
// alternativ nicht möglich
// RegExp.input="Otto"
// dann kein detailliertes Suchmuster
// Option g alternativ kodierbar per
// RegExp.multiline=true;

var ergebnis_kette = zu_durchsuchende_kette.replace(such_muster,"Heinrich");

```

`.replaceAdjacentText()` Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern



DOM wird nicht geändert

Beispiel:

```
var PlainText = " Neuer Text ";
ID_Div.replaceAdjacentText("afterBegin", PlainText);
```

```
<DIV ID="ID_Div">
  Test
</DIV>
```

.replaceChild()

Kind-Objekt ersetzen durch ein Objekt
 ersetzende Objekt muss per Methode `.createElement()` erzeugt worden sein
 Sichtbarkeit erst wenn Ende-Tag geparkt wurde
 DOM wird geändert

Beispiel:

```
<HEAD>
<SCRIPT>
  function Ersetze()
  {
    var KindZeigerAufDivText = ID_Div.children(0);
    var RetteInnerHTML = KindZeigerAufDivText.innerHTML;

    // prüfen auf Tag im Div-Text
    if (KindZeigerAufDivText.tagName=="B")
    {
      // Bold-Tag <B>gefunden, also I-Tag erzeugen
      var ZeigerAufNeuenSchriftStilTag =document.createElement("I");

      // komplettes ersetzen von Div-Text,
      ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
      ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
    else
    {
      // keinen Bold-Tag <B>gefunden
      var ZeigerAufNeuenSchriftStilTag =document.createElement("B");

      // komplettes ersetzen von Div-Text,
      ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
      ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
  }
</SCRIPT>
</HEAD>
<BODY>
  <DIV ID="ID_Div" onclick=" Ersetze()">
    Klicke für den Wechseln des <B>Schriftstils<B>
  </DIV>
</BODY>
```

.replaceData()

Teilkette in einem Objekt ersetzen

.replaceNode()

Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
 sichtbar erst mit parsen des Endetags
 DOM wird geändert

Beispiel:

```
<SCRIPT>
  function Ersetze()
  {
    var RetteInnerHTML = Liste.innerHTML;
    var ZeigerAufNeuenKnoten = document.createElement("OL");
    Liste.replaceNode(ZeigerAufNeuenKnoten);
    ZeigerAufNeuenKnoten.innerHTML = RetteInnerHTML;
  }
</SCRIPT>
<UL ID = "Liste" >
  <LI>Listeneintrag 1
  <LI>Listeneintrag 2
  <LI>Listeneintrag 3
  <LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Ersetze" onclick = "Ersetze()">
```

.reset()

löst für Formular das Event onreset aus, dessen Eventhandler die Reset-Aktion beinhaltet, die gestartet wird
 VOR dem Reset der Elemente (Browser setzt zurück)



Eventhandler muss liefern: return true, für Ausführung des Reset durch Browser
 return false; für Nicht-Ausführung des Reset durch Browser

Beispiel: Man beachte, dass NAME-Attribut für alle zu sendenden bzw. rückzusetzenden Elemente **Pflicht** ist !!

```
<HTML>
<HEAD>
<SCRIPT>
function EventHandler_AktionVorReset()
{
    // ein Hinweis im Textarea anzeigen
    ID_Textarea.value += "Formular zuruecksetzen ???";

    // wirklich rücksetzen ???
    return( confirm("Wirklich rücksetzen ?"));
    // true so Reset durch Browser ausführen lassen
    // false so kein Reset durch Browser
}
</SCRIPT>
</HEAD>
<BODY>
<DIV>
<FORM NAME="Formular" onreset="EventHandler_AktionVorReset();">
    <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
    <BR>
    <BUTTON onclick="form.reset();">OnReset ausloesen</BUTTON>
    <BR><BR>
    <INPUT TYPE="reset" VALUE="oder hiermit zuruecksetzen"
        onclick="form.reset()"
    >
</FORM>
</DIV>
<TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>
```

.resetElement() alle Timeline-Einstellungen zum Element aufheben (egal ob Element aktiv oder nicht)
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
    REPEATDUR="indefinite"
    onbegin="ID_Button1.disabled=false; ID_Button2.disabled=true;"
>
    <DIV ID="ID_Div1"
        CLASS="time_line_klasse"
        BEGIN="0"
        DUR="5"
    >
        Erste Zeile
    </DIV>
    <DIV ID="ID_Div2"
        CLASS="time_line_klasse"
        BEGIN="5"
        DUR="5"
    >
        Zweite Zeile
    </DIV>
</t:EXCL>
<BR>
<BUTTON ID="ID_Button1"
    onclick=" ID_Excl.endElementAt(5);
              alert('Time-Container endet in 5 Sekunden');
              ID_Button1.disabled=true;
              ID_Button2.disabled=false;
              "
>
```



```

        Ende der Timeline in 5 Sekunden
    </BUTTON>
    <BUTTON
        ID="ID_Button2"
        onclick="ID_Excl.resetElement();
                alert('Alle Einstellungen zuruecksetzen');
                ID_Button1.disabled=false;
                ID_Button2.disabled=true;
        "
    >
        Reset
    </BUTTON>
</BODY>
</HTML>

```

.resizeBy() Fenstergröße um Pixeldifferenz verändern
 Fenstergröße auf weniger als 100 x 100 Pixel ist nicht möglich
 nicht bei Dialog-Fenster: Dafür dialogHeight, dialogWidth, dialogTop, dialogLeft benutzen
 funktioniert nur, wenn open-Merkmal resizable=yes kodiert wurde
 siehe Objekt window

.resizeTo() Fenstergröße neu dimensionieren in Pixel
 Fenstergröße auf weniger als 100 x 100 Pixel ist nicht möglich
 nicht bei Dialog-Fenster: Dafür dialogHeight, dialogWidth, dialogTop, dialogLeft benutzen
 funktioniert nur, wenn open-Merkmal resizable=yes kodiert wurde
 siehe Objekt window

Beispiel für Fensterauflösung ändern:

```
javascript:window.resizeTo(640,480); javascript:window.resizeTo(800,600); javascript:window.resizeTo(1024,768)
```

Beispiel für sich aufblasendes Fenster von 100x100 bis auf 640x480

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
    var start_hoehe=100;
    var start_breite=100;
    var max_hoehe=480;
    var max_breite=640;
    var aktuelle_hoehe=start_hoehe;
    var aktuelle_breite=start_breite;
    var y=5;
    var TimerID=null;
    var fenster;

function start()
{
    fenster=window.open("", "", "scrollbars");

    if ( document.layers || document.all)
    {
        fenster.resizeTo(start_breite,start_hoehe);
        fenster.moveTo(0,0);
        blasen();
    }
    else
    {alert("Weder Netscape noch Microsoft erkannt !")};
}

function blasen()
{
    if (aktuelle_hoehe>=max_hoehe)
    {x=0;}

    fenster.resizeBy(5,y);
    aktuelle_hoehe+=5;
    aktuelle_breite+=5;

    if (aktuelle_breite>=max_breite)
    {
        alert("Maximal aufgeblasen !");
        fenster.close();
        x=5;
    }
}

```



```

        TimerID=null;
    }
    else
    {TimerID=setTimeout("blasen()",50);}
}
//-->
</SCRIPT>
</HEAD>
<BODY>
<A      HREF="javascript:start()"
        onMouseOver="javascript:window.status='Oeffne Fenster';return true;" // Text nach Statuszeile
        onMouseout="javascript:window.status=";" // Statuszeile löschen
>Oeffne NEUES Fenster mit 100x100 Pixel und blase es auf 640x480 Pixel !
</A>
</BODY>
</HTML>

```

`.resume()` deprecated
Pause auf der Timeline beenden

`.resumeElement()` Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben
falls Element nicht pausiert, passiert nichts
ersetzt die Methode `resume()`, da sie deprecated ist und nicht mehr verwendet werden darf
per Eigenschaft `.isActive` das Element auf Aktivsein prüfen
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
.TextAufHighLight{ color:#CCCCC; font-weight:bold;}
</STYLE>
<SCRIPT>
function PauseAufheben()
{
    ID_Table.resumeElement();
    ID_Button1.disabled = false;
    ID_Button2.disabled = true;
}

function Pausieren()
{
    ID_Table.pauseElement();
    ID_Button1.disabled = true;
    ID_Button2.disabled = false;
}
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<TABLE ID="ID_Table"
        BORDER="1"
        CLASS="time_line_klasse"
        DUR="8s"
        REPEATCOUNT="indefinite"
        TIMECONTAINER="SEQ"
        TIMEACTION="none"
>
<TBODY>
<TR>
<TH WIDTH="120">Eins</TH>
<TH WIDTH="50">Zwei</TH>
<TH WIDTH="40">Drei</TH>
</TR>
<TR
        ID="ID_TR1"
        CLASS="time_line_klasse"
        TIMEACTION="class: TextAufHighLight "
        DUR="2s"
>
<TD>EinsA</TD>
<TD>ZweiA</TD>
<TD>DreiA</TD>
</TR>
<TR
        ID="ID_TR2"

```



```

        CLASS="time_line_klasse"
        TIMEACTION="class: TextAufHighLight "
        DUR="2s"
    >
        <TD>EinsB</TD>
        <TD>ZweiB</TD>
        <TD>DreiB</TD>
    </TR>
</TBODY>
</TABLE>
<BR>
<BUTTON        ID="ID_Button1"
                onclick="Pausieren();"
>
    pausieren
</BUTTON>
<BUTTON        ID="ID_Button2"
                DISABLED="true"
                onclick="PauseAufheben();"
>
    Pause aufheben
</BUTTON>
</BODY>
</HTML>

```

.reverse() Reihenfolge der Feldelemente physisch umkehren, also

<u>alt</u>	<u>neu</u>
1. Element	letztes Element
letztes Element	1. Element

Feld hat die Objektklasse Script-Objekt Array

Beispiel:

```

var Feld1 = new Array(0,1,2,3,4);           // Feld 1 ist Zeiger vor reverse
var Feld2 = Feld1.reverse();                 // Feld 2 ist Zeiger nach reverse, Feld1 ist ungültig
alert(Feld2.join()); // "4,3,2,1,0"

```

.round() zahl auf ganz runden; ab >=5 aufwärts
Bsp: 0,5 ergibt 1
siehe Script-Objekt Math

.save() UserDataStore (User-Cache) speichern per .style.userData Behavior

Beispiel:

```

<HTML>
<HEAD>
<STYLE>
    .user_data_speicher_klasse {behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
    // Cachename festlegen
    // es können diverse Cachennamen definiert und somit Versionen von Cache
    // verwaltet werden
    var FreierCacheName = "InputCache";

    // Cache-Attribut festlegen
    // es können diverse Attribute definiert und somit Versionen von Input-Daten
    // verwaltet werden
    var FreiesCacheAttribut = "InputCacheAttribut";

    // zu cachende Daten referenzieren
    var InputDatenObjekt = ID_Formular.ID_Input;

    function InputSichern()
    {
        // ++++++ Zeitstempel ++++++
        var ZeitpunktJetzt = new Date();
        var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

        // Zeitstempel festlegen: Ab jetzt + 20 Minuten
        var Zeitstempel = ZeitpunktJetztInMinuten + 20;

        // und als UTC-Format erzeugen
        var ZeitstempelUTC = Zeitstempel.toUTCString();

        // und Zeitstempel dem Input-Objekt verpassen
        ID_Input.expires = ZeitstempelUTC;
    }

```




```

// ++++++ Daten chachen ++++++
// aktuelle Daten holen laut Input-Objekt
var InputDaten = InputDatenObjekt.value;

// Attribut instanzieren und mit Daten füllen
InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

// und Cache saveen
InputDatenObjekt.save(FreierCacheName);
}

function InputLaden()
{
    // Cache laden
    InputDatenObjekt.load(FreierCacheName);

    // und Daten zum Attribut laut Sicherung lesen
    var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
}

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

.ScriptEngine() Sprache der gerade benutzten Scriptmaschine im Internet Explorer

"JScript"	Microsoft JScript
"VBA"	Microsoft Visual Basic for Applications
"VBScript"	Microsoft Visual Basic Scripting Edition

Beispiel:

```

function Anzeigen()
{
    var Kette = "";

    Kette += "Aktuelle Maschine = " + ScriptEngine();
    Kette += " mit Hauptversion " + ScriptEngineMajorVersion();
    Kette += " mit Unterversion " + ScriptEngineMinorVersion();
    Kette += " mit Buildnummer " + ScriptEngineBuildVersion();

    alert(Kette);
}

```

.ScriptEngineBuildVersion() Buildnummer der gerade benutzten Scriptmaschine im Internet Explorer

Beispiel:

```

function Anzeigen()
{
    var Kette = "";

    Kette += "Aktuelle Maschine = " + ScriptEngine();
    Kette += " mit Hauptversion " + ScriptEngineMajorVersion();
    Kette += " mit Unterversion " + ScriptEngineMinorVersion();
    Kette += " mit Buildnummer " + ScriptEngineBuildVersion();

    alert(Kette);
}

```

.ScriptEngineMajorVersion() Hauptversion der gerade benutzten Scriptmaschine im Internet Explorer

Beispiel:

```

function Anzeigen()
{
    var Kette = "";

    Kette += "Aktuelle Maschine = " + ScriptEngine();
    Kette += " mit Hauptversion " + ScriptEngineMajorVersion();
}

```



	<pre> Kette += " mit Unterversion " + ScriptEngineMinorVersion(); Kette += " mit Buildnummer " + ScriptEngineBuildVersion(); alert(Kette); } </pre>
<code>.ScriptEngineMinorVersion()</code>	Unterversion der gerade benutzten Scriptmaschine im Internet Explorer
Beispiel:	<pre> function Anzeigen() { var Kette = ""; Kette += "Aktuelle Maschine = " + ScriptEngine(); Kette += " mit Hauptversion " + ScriptEngineMajorVersion(); Kette += " mit Unterversion " + ScriptEngineMinorVersion(); Kette += " mit Buildnummer " + ScriptEngineBuildVersion(); alert(Kette); } </pre>
<code>.scroll()</code>	<p>deprecated und zu ersetzen durch <code>.scrollBy()</code> bzw. <code>.scrollTo()</code></p> <p>linke obere Ecke des Dokumentes im Fenster verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters</p> <p>Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrolleisen etc.</p> <p>sinnvoll, wenn automatische Anzeige von Scrolleisen verhindert wurde</p> <p>Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters</p> <p>siehe Objekt window</p>
<code>.scrollBy()</code>	<p>ersetzt <code>.scroll()</code></p> <p>linke obere Ecke des Dokumentes im Fenster verschieben um Pixeldifferenz bezüglich linker oberer Ecke des Fensters</p> <p>Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrolleisen etc.</p> <p>sinnvoll, wenn automatische Anzeige von Scrolleisen verhindert wurde</p> <p>Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters</p> <p>siehe Objekt window</p>
<code>.scrollIntoView()</code>	<p>Objekt derart scrollen, dass es im Fenster für User sichtbar wird</p> <p>Objekt muss an sich schon renderbar sein</p> <p>true Default Obere Kante des Objekt bis an den oberen Fensterrand scrollen</p> <p>false Untere Kante des Objektes an den unteren Fensterrand scrollen</p>
Beispiel:	<pre> var FeldAllerPTags = document.all.tags("P"); FeldAllerPTags[4].scrollIntoView(true); </pre>
<code>.scrollTo()</code>	<p>ersetzt <code>.scroll()</code></p> <p>linke obere Ecke des Dokumentes im Fenster auf Pixelposition bezüglich linker oberer Ecke des Anzeigebereiches des Fensters setzen</p> <p>Anzeigebereich: innerhalb des Fensterrahmens und abzüglich aller Fenster-Elemente wie Menüleiste, Scrolleisen etc.</p> <p>sinnvoll, wenn automatische Anzeige von Scrolleisen verhindert wurde</p> <p>Koordinaten-Ursprung (0,0) liegt in der linken oberen Ecke des Fensters</p> <p>siehe Objekt window</p>

Beispiel für Dokument vertikales Scrollen:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function scrollen(pixel_anzahl)
    {
        for (i=1; i<pixel_anzahl; i++)
        {window.scrollTo(0,i);}
    }
-->
</SCRIPT>
</HEAD>

<BODY>
....
<INPUT TYPE=button

```



```

        VALUE="Start scrollen!"
        onclick="scrollen(100);"
    >
</BODY>
</HTML>

```

`.search()` Suche in einem String oder String-literal per RegExp-Objekt (siehe dort Methode `.exec()`)
siehe Script-Objekt String

Beispiel 1:

```

var Kette          = "The rain in Spain falls mainly in the plain.";
var RegExpAusdruck = /falls/i;
var Wert           = Kette.search(RegExpAusdruck);

```

Beispiel 2:

```

var zu_durchsuchende_kette = "Otto";
var such_muster             = /(\wOtto)/g; // vom Typ RegExp, also regulärer Ausdruck
var ergebnis              = zu_durchsuchende_kette.search(such_muster);

```

`.seekActiveTime()` aktives Element animieren ab einem Zeitpunkt auf der Timeline
wenn Element nicht aktiv, so Fehler erzeugt
per Eigenschaft `.isActive` das Element auf Aktivsein prüfen
alle Media-Typen für Element zulässig
siehe Eigenschaft `.canSeek`
siehe Objekt `currTimeState` und Behavior `.style.time2`

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    function Suchen()
    {
        // prüfen ob Element nicht aktiv ist
        if (!ID_Video.currTimeState.isActive)
        {
            // nicht aktiv, also starten
            ID_Video.beginElement();
        }
        else
        {
            // prüfen des Eingabewertes
            if( ( isFinite(ID_Input.value) )
                && (ID_Input.value <= ID_Video.mediaDur )
                && (ID_Input.value > 0)
            )
            {
                // ist okay, also ab Zeitpunkt animieren
                ID_Video.seekActiveTime(ID_Input.value);
            }
            else
            {
                // fehlerhaft
                alert( "Fehler ! Sekunden-Wert muss > 0 und < Wert laut mediaDur = "
                    + ID_Video.mediaDur
                    + " sein"
                );
                ID_Input.focus();
            }
        }
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
          CLASS="time_line_klasse"
          DUR=".01"
          REPEATCOUNT="indefinite"
          FILL="hold"
          onrepeat="innerText=parseInt(ID_Video.currTimeState.activeTime);"
    >
        0
    </SPAN>

```



```

<BR>
<t:VIDEO ID="ID_Video"
SRC="test.avi"
STYLE="width:175px; height:150px;"
onmediacomplete="ID_Span2.innerText= ID_Video.mediaDur;"
>
</t:VIDEO>
<BR>
Dauer des AVI:
<SPAN ID="ID_Span2"></SPAN>
&nbsp;  Sekunden
<BR>
setze seekActiveTime:
<INPUT NAME="ID_Input"
TYPE="text"
VALUE=""
SIZE="3"
>&nbsp;  Sekunden
<BR>
<BUTTON ID="ID_Button1"
onclick="Suchen();"
>
Klick fuer Seek
</BUTTON>
<BUTTON ID="ID_Button2"
onclick="ID_Video.beginElement()"
>
Restart
</BUTTON>
</BODY>
</HTML>

```

.seekSegmentTime() aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
 ohne Wiederholung der Animation
 wenn Element nicht aktiv, so Fehler erzeugt
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 Software-Player zum Mediatyp des Elementes wird nicht davon beeinflusst
 nicht alle Media-Typen für Element zulässig
 wenn unzulässig, so kein Fehler
 siehe Eigenschaft .canSeek
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
function Suchen()
{
    // prüfen ob Element nicht aktiv ist
    if (!ID_Video.currTimeState.isActive)
    {
        // nicht aktiv, also starten
        ID_Video.beginElement();
    }
    else
    {
        // prüfen des Eingabewertes
        if( ( isFinite(ID_Input.value) )
        && (ID_Input.value <= ID_Video.segmentDur )
        && (ID_Input.value > 0)
        )
        {
            // ist okay, also ab Zeitpunkt animieren
            ID_Video.seekSegmentTime(ID_Input.value);
        }
        else
        {
            // fehlerhaft
            alert( "Fehler ! Sekunden-Wert muss > 0 und < Wert laut segmentDur = "
            + ID_Video.segmentDur
            + " sein"
            );
        }
    }
}

```



```

    );
    ID_Input.focus();
  }
}
</SCRIPT>
</HEAD>
<BODY>
  <SPAN ID="ID_Span1"
    CLASS="time_line_klasse"
    DUR=".01"
    REPEATCOUNT="indefinite"
    FILL="hold"
    onrepeat="innerText=parseInt(ID_Video.currTimeState.activeTime);"
  >
    0
  </SPAN>
  <BR>
  <t:VIDEO ID="ID_Video"
    SRC="test.avi"
    STYLE="width:175px; height:150px;"
    onmediacomplete="ID_Span2.innerText= ID_Video.segmentDur;"
  >
  </t:VIDEO>
  <BR>
  Dauer des AVI:
  <SPAN ID="ID_Span2"></SPAN>
  &nbsp;Sekunden
  <BR>
  setze seekSegmentTime:
  <INPUT NAME="ID_Input"
    TYPE="text"
    VALUE=""
    SIZE="3"
  >&nbsp;Sekunden
  <BR>
  <BUTTON ID="ID_Button1"
    onclick="Suchen();"
  >
    Klick fuer Seek
  </BUTTON>
  <BUTTON ID="ID_Button2"
    onclick=" ID_Video.beginElement()"
  >
    Restart
  </BUTTON>
</BODY>
</HTML>

```

.seekTo() aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline
 einschliesslich möglicher Wiederholungen der Animation
 wenn Element nicht aktiv, so Fehler erzeugt
 per Eigenschaft .isActive das Element auf Aktivsein prüfen
 nicht alle Media-Typen für Element zulässig
 wenn unzulässig, so kein Fehler
 siehe Eigenschaft .canSeek
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
  .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
  function Suchen()
  {
    // prüfen ob Element nicht aktiv ist
    if (!ID_Video.currTimeState.isActive)
    {
      // nicht aktiv, also starten
      ID_Video.beginElement();
    }
    else
  }

```



```

        {
            // prüfen des Eingabewertes
            if( ( isFinite(ID_Input.value) )
                && (ID_Input.value <= ID_Video.segmentDur )
                && (ID_Input.value > 0)
            )
            {
                // ist okay, also ab Zeitpunkt animieren
                ID_Video.seekTo(1,ID_Input.value);
            }
            else
            {
                // fehlerhaft
                alert( "Fehler ! Sekunden-Wert muss > 0 und < Wert laut segmentDur = "
                    + ID_Video.segmentDur
                    + " sein"
                );
                ID_Input.focus();
            }
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Video.currTimeState.activeTime);"
    >
        0
    </SPAN>
    <BR>
    <t:VIDEO ID="ID_Video"
        SRC"test.avi"
        STYLE="width:175px; height:150px;"
        onmediacomplete="ID_Span2.innerText= ID_Video.segmentDur;"
    >
    </t:VIDEO>
    <BR>
    Dauer des AVI:
    <SPAN ID="ID_Span2"></SPAN>
    &nbsp;&nbsp;&nbsp;Sekunden
    <BR>
    setze seekTo:
    <INPUT TYPE="text"
        VALUE=""
        NAME="ID_Input"
        SIZE="3"
    >&nbsp;&nbsp;&nbsp;Sekunden
    <BR>
    <BUTTON ID="ID_Button1" onclick="Suchen();">
        Klick fuer Seek
    </BUTTON>
    <BUTTON ID="ID_Button2" onclick=" ID_Video.beginElement();">
        Restart
    </BUTTON>
</BODY>
</HTML>

```

.seekToFrame()

Frame eines aktiven Elementes auf der Timeline anwählen
wenn Element nicht aktiv, so Fehler erzeugt
per Eigenschaft .isActive das Element auf Aktivsein prüfen
nicht alle Media-Typen für Element zulässig
wenn unzulässig, so kein Fehler
siehe Eigenschaft .canSeek
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }

```



```

</STYLE>
<SCRIPT>
    var FrameAnzahl=600;

    function Suchen()
    {
        // prüfen ob Element nicht aktiv ist
        if (!ID_Video.currTimeState.isActive)
        {
            // nicht aktiv, also starten
            ID_Video.beginElement();
        }
        else
        {
            // prüfen des Eingabewertes
            if( ( isFinite(ID_Input.value) )
                && (ID_Input.value < FrameAnzahl )
                && (ID_Input.value > 0)
            )
            {
                // ist okay, also ab Zeitpunkt animieren
                ID_Video.seekToFrame(ID_Input.value);
            }
            else
            {
                // fehlerhaft
                alert( "Fehler ! Frame-Wert muss > 0 und < "
                    + FrameAnzahl
                    + " sein"
                );
                ID_Input.focus();
            }
        }
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span1"
        CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Video.currentFrame);"
    >
        0
    </SPAN>
    <BR>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        STYLE="width:175px; height:150px;"
        onmediacomplete="ID_Span2.innerText= ID_Video.mediaDur;"
    >
    </t:VIDEO>
    <BR>
    Dauer des AVI:
    <SPAN ID="ID_Span2"></SPAN>
    &nbsp;Sekunden
    <BR>
    setze seekToFrame:
    <INPUT TYPE="text"
        VALUE=""
        NAME="ID_Input"
        SIZE="3"
    >&nbsp;Sekunden
    <BR>
    <BUTTON ID="ID_Button1" onclick="Suchen();">
        Klick fuer Seek
    </BUTTON>
    <BUTTON ID="ID_Button2" onclick=" ID_Video.beginElement();">
        Restart
    </BUTTON>
</BODY>
</HTML>

```



.segmentTimeToActiveTime() Wert der Segment-Timeline des Elements in den korrespondierenden Wert der aktiven Timeline des Elements konvertieren
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
window.onload = Rekursion;    // ohne () kodieren

function Rekursion()
{ window.setInterval(Anzeige, 100);}

function Anzeige()
{
    ID_Span1.innerHTML = "segmentTimeToActiveTime: "
                        + (ID_Animation.segmentTimeToActiveTime(
                            ID_div.currTimeState.activeTime
                        ));
    ID_Span1.innerHTML = "activeTime: "
                        + (ID_Animation.currTimeState.activeTime);
}
</SCRIPT>
</HEAD>
<BODY>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<t:ANIMATEMOTION ID="ID_Animation"
TARGETELEMENT="ID_div"
TO="250,0"
DUR="3"
AUTOREVERSE="true"
>
</t:ANIMATEMOTION>
<SPAN ID="ID_Span1">
segmentTimeToActiveTime:
</SPAN>
<BR>
<SPAN ID="ID_Span2">
activeTime:
</SPAN>
</BODY>
</HTML>
```

.segmentTimeToSimpleTime() Wert der Segment-Timeline eines Elements in den korrespondierenden Wert der Simple-Timeline des Elements konvertieren
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
window.onload = Rekursion;    // ohne () kodieren

function Rekursion()
{ window.setInterval(Anzeige, 100);}

function Anzeige()
{
    ID_Span1.innerHTML = "segmentTimeToSimpleTime: "
                        + (ID_Animation.segmentTimeToSimpleTime(
                            ID_div.currTimeState.activeTime
                        ));
}
```




```

        ID_Span1.innerHTML = " simpleTime: "
        + (ID_Animation.currTimeState.simpleTime);
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div"
        CLASS="time_line_klasse"
        STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
    >
    </DIV>
    <t:ANIMATEMOTION ID="ID_Animation"
        TARGETELEMENT="ID_div"
        TO="250,0"
        DUR="3"
        AUTOREVERSE="true"
    >
    </t:ANIMATEMOTION>
    <SPAN ID="ID_Span1">
        segmentTimeToSimpleTime:
    </SPAN>
    <BR>
    <SPAN ID="ID_Span2">
        simpleTime:
    </SPAN>
</BODY>
</HTML>

```

.select() Bereich des Input-Objektes im Formular markieren
setzt **nicht** den Focus (dafür Methode .focus() verwenden)

.select() Objekt selektieren
z.B. Textbereich: wird hervorgehoben
ControlRange: es wird Rechteck-Rahmen erzeugt
nur unter Windows 32-Bit

Beispiel:

```

<SCRIPT LANGUAGE="JScript" FOR=document EVENT=onclick >
    var TextBereich = document.body.createTextRange();
    TextBereich.moveToPoint(window.event.x, window.event.y);
    TextBereich.expand("word");
    TextBereich.select();
</SCRIPT>

```

.select() Selektion eines Textranges (Textbereich, Objekt textrange) oder ControlRange (Control-Elemente)
nur unter Windows 32-Bit
siehe Objekt document.selection
Methoden .createTextRange() .createControlRange() und .createRange()

Beispiel 1 Textrange (Textbereich) erzeugen und den Inhalt markieren, also selektieren

```

function ErzeugeUndSelektiereTextRange()
{
    var TextBereich = document.body.createTextRange();
    TextBereich.findText("Testtext");
    TextBereich.select();
}

```

Beispiel 2 Controlrange erzeugen und den Inhalt markieren, also selektieren

```

function ControlRangeErzeugenUndSelektieren()
{
    var ControlRangeObjekt = document.body.createControlRange();
    ControlRangeObjekt.add(document.all.zeiger_auf_control_element);
    ControlRangeObjekt.select();
}

```

Hinweis: Elternobjekte mit Textrange sind

body Objekt
button Objekt
textarea Objekt
input text Objekt
selection Objekt
(nur wenn ein Text selektiert wurde (mit oder ohne HTML))



können weitere HTML-Elemente enthalten, die ebenfalls Textbereiche besitzen können

`.setActive()` Objekt für die Eventdurchreichung aktivieren
aber ohne es zu fokussieren
und ohne es scrollbar zu machen

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
    var ID_Fenster;

    function FensterErzeugen()
    {
        ID_Fenster = window.open( "/test /test.htm",
                                " ID_Fenster",
                                "top=10px,left=480px,height=375px,width=200px,resizable=1"
                                );
        this.focus();
    }

    function ButtonAktivieren()
    {window.parent.ID_Fenster.ID_Button.setActive();}
</SCRIPT>
</HEAD>
<BODY onload="FensterErzeugen();">
    <BUTTON ID="ID_Button" onclick="ButtonAktivieren();">
        Button aktivieren
    </BUTTON>
</BODY>
</HTML>
```

`.setActive()` playItem Objekt als aktiven Track setzen und als solchen in der Behavior-Collection `.style.time2.playList` registrieren
verändert `.isActive`
Track entspricht playItem Objekt (Playlisten-Eintrag)
Trackliste liegt in der Behavior-Collection `.style.time2.playList` als Playliste:
Diese ist nur referenzierbar, solange das Elternobjekt des Elementes auf der Eltern-Timeline aktiv ist, also das Elternobjekt noch nicht auf seiner Timeline geendet ist.
Advanced Stream Redirector (ASX) –Datei: Playlisten-Datei
aktiver Track in der Liste: wird abgespielt
Track in der Liste auf aktiv setzen: Abspielen des Tracks sofort starten

Beispiel 1:

`object.playList.item(index).setActive()`

Index Integer, ab 0

Beispiel 2:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
<SCRIPT LANGUAGE="JScript">
    function ButtonUpdate()
    {
        if (ID_Media.currTimeState.isActive)
        {
            ID_Button1.disabled=true;
            ID_Button2.disabled=false;
            ID_Button3.disabled=false;
            ID_Button4.disabled=false;
        }
        else
        {
            ID_Button1.disabled=false;
            ID_Button2.disabled=true;
            ID_Button3.disabled=true;
            ID_Button4.disabled=true;
        }
    }

    function AnzeigeUpdate()
```



```

    {
        ID_Span1.innerText = "Titel: " + ID_Media.playlist.activeTrack.title;
        ID_Span3.innerText = "Autor: " + ID_Media.playlist.activeTrack.author;
        ID_Span3.innerText = "Abstract: " + ID_Media.playlist.activeTrack.abstract;
        ID_Span4.innerText = "Copyright: " + ID_Media.playlist.activeTrack.copyright;
        ID_Span5.innerText = "Filename: " + ID_Media.playlist.activeTrack.src;
        ID_Span6.innerText = "Banner: " + ID_Media.playlist.activeTrack.Banner;
        ID_Span7.innerText = "Banner Abstract: " + ID_Media.playlist.activeTrack.BannerAbstract;
        ID_Span8.innerText = "Banner MoreInfo: " + ID_Media.playlist.activeTrack.BannerMoreInfo;
    }

function AnzeigeLoeschen()
{
    ID_Span1.innerText = "Titel: ";
    ID_Span2.innerText = "Autor: ";
    ID_Span3.innerText = "Abstract: ";
    ID_Span4.innerText = "Copyright: ";
    ID_Span5.innerText = "Filename: ";
    ID_Span6.innerText = "Banner: ";
    ID_Span7.innerText = "Banner Abstract: ";
    ID_Span8.innerText = "Banner MoreInfo: ";
}
</SCRIPT>
</HEAD>
<BODY onload="ButtonUpdate()">

    <t:MEDIA ID="ID_Media"
        SRC="test.asx"
        BEGIN="indefinite"
        TIMEACTION="visibility"
        onend="ButtonUpdate();"
        ontrackchange="AnzeigeUpdate();"
        onmediacomplete="ButtonUpdate();AnzeigeUpdate();"
    >
    </t:MEDIA>

    <SPAN ID="ID_Span1">Titel:</SPAN>
    <BR>
    <SPAN ID="ID_Span2">Autor:</SPAN>
    <BR>
    <SPAN ID="ID_Span3">Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span4">Copyright:</SPAN>
    <BR>
    <SPAN ID="ID_Span5">Filename:</SPAN>
    <BR>
    <SPAN ID="ID_Span6">Banner:</SPAN>
    <BR>
    <SPAN ID="ID_Span7">Banner Abstract:</SPAN>
    <BR>
    <SPAN ID="ID_Span8">Banner MoreInfo:</SPAN>

    <BR>
    <BUTTON ID="ID_Button1"
        onclick="ID_Media.beginElement(); ButtonUpdate();"
    >
        Start
    </BUTTON>
    <BUTTON ID="ID_Button2"
        onclick="ID_Media.playlist.nextTrack();"
    >
        naechster Track
    </BUTTON>
    <BUTTON ID="ID_Button3"
        onclick="ID_Media.playlist.prevTrack();"
    >
        vorhergehender Track
    </BUTTON>
    <BUTTON ID="ID_Button4"
        onclick="ID_Media.endElement(); AnzeigeLoeschen();"
    >
        Stop
    </BUTTON>
</BODY>

```



</HTML>

.setActive() Fenster als aktiv setzen (also auch für die Eventdurchreichung aktivieren)
 aber **ohne** es zu fokussieren
 und **ohne** es scrollbar zu machen
 siehe Objekt window

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
  var ID_Fenster;

  function FensterErzeugen()
  {
    ID_Fenster = window.open( "/test /test.htm",
                              " ID_Fenster",
                              "top=10px,left=480px,height=375px,width=200px,resizable=1"
                              );
    this.focus();
  }

  function ButtonAktivieren()
  {window.parent.ID_Fenster.ID_Button.setActive();}
</SCRIPT>
</HEAD>
<BODY onload="FensterErzeugen();">
  <BUTTON ID="ID_Button" onclick="ButtonAktivieren();">
    Button aktivieren
  </BUTTON>
</BODY>
</HTML>
```

.setAttribute() Wert von vorhandenem Attribut setzen
 wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
 DOM wird nur bei Erzeugung geändert

Beispiel:

```
<HTML>
<HEAD>
<STYLE>
  .user_data_speicher_klasse { behavior:url(#default#userData);}
</STYLE>
<SCRIPT>
  // Cachname festlegen
  // es können diverse Cachennamen definiert und somit Versionen von Cache
  // verwaltet werden
  var FreierCacheName = "InputCache";

  // Cache-Attribut festlegen
  // es können diverse Attribute definiert und somit Versionen von Input-Daten
  // verwaltet werden
  var FreiesCacheAttribut = "InputCacheAttribut";

  // zu cachende Daten referenzieren
  var InputDatenObjekt = ID_Formular.ID_Input;

  function InputSichern()
  {
    // ++++++ Zeitstempel ++++++
    var ZeitpunktJetzt = new Date();
    var ZeitpunktJetztInMinuten = ZeitpunktJetzt.getMinutes();

    // Zeitstempel festlegen: Ab jetzt + 20 Minuten
    var Zeitstempel = ZeitpunktJetztInMinuten + 20;

    // und als UTC-Format erzeugen
    var ZeitstempelUTC = ZeitpunktJetzt.toUTCString();

    // und Zeitstempel dem Input-Objekt verpassen
    ID_Input.expires = ZeitstempelUTC;

    // ++++++ Daten chachen ++++++
    // aktuelle Daten holen laut Input-Objekt
    var InputDaten = InputDatenObjekt.value;
```



```

        // Attribut instanzieren und mit Daten füllen
        InputDatenObjekt.setAttribute(FreiesCacheAttribut, InputDaten);

        // und Cache saveen
        InputDatenObjekt.save(FreierCacheName);
    }

    function InputLaden()
    {
        // Cache laden
        InputDatenObjekt.load(FreierCacheName);

        // und Daten zum Attribut laut Sicherung lesen
        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
    }

</SCRIPT>
</HEAD>
<BODY>
    <FORM ID="ID_Formular">
        <INPUT ID="ID_Input"
            CLASS="user_data_speicher_klasse"
            TYPE="text"
        >
        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
        <INPUT TYPE="button" VALUE="laden der Input-Daten" onclick="InputLaden()">
    </FORM>
</BODY>
</HTML>

```

`.setAttributeNode()` Attribut einem Knoten zuweisen und Referenz liefern
DOM wird geändert

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
    function Hinzufuegen()
    {
        // Attribut mit Wert erzeugen
        var ZeigerAufAttribut = document.createAttribute("title");
        ZeigerAufAttribut.value = "Tooltip-Text ";

        // Attribut als Knoten erzeugen
        var AttributKnoten = ID_Div.setAttributeNode(ZeigerAufAttribut);
    }
</SCRIPT>
</HEAD>
<BODY onload=" Hinzufuegen();">
    <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

`.setCapture()` Maus-Überwachung einschalten für ein Objekt
Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,
onmouseover und onmouseout.

ab IE 5.5

Hinweis: ausschalten per Methode `.releaseCapture()`

true Default. Überwachung durch Kinder nicht möglich

false Überwachung durch Kinder möglich

Beispiel:

```

<BODY onload="ID_Div.setCapture();"
    onclick="document.releaseCapture();"
>
    <DIV ID="ID_Div"
        onmousemove="ID_Textarea.value = event.clientX + event.clientY;"
        onlosecapture="alert(event.srcElement.id + ' hat keine Mausueberwachung mehr');"
    >
        <TEXTAREA ID="ID_Textarea" COLS=2>Test</TEXTAREA>
    </DIV>
</BODY>

```

`.setData()` Clipboard füllen, also Daten dort ablegen
wenn Clipboard nicht leer so immer anhängen



<code>.setDate()</code>	Tag des Monats in lokaler Zeit setzen wenn Tag > Anzahl Tage im Monat, so erfolgt automatisch Monatswechsel und Korrektur aller Angaben (inklusive Jahr) Script-Objekt Date Integer 1 bis 31
<code>.setEndPoint()</code>	Textbereichanfang bzw. -ende von 2 Textbereichen synchronisieren per <code>textrange</code> Objekt nur unter Windows 32-Bit
<code>.setExpression()</code>	Wert definieren, der als Ausdruck für die Methode <code>.getExpression()</code> zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form <code>objekt.style.eigenschaft.</code> dient Ausdruck nur als Script kodierbar DOM wird nicht geändert

In nachfolgenden Beispielen wurde aus Platzgründen die Zeichenkette von `STYLE` umgebrochen, was eigentlich nicht zulässig ist, und es sind nicht alle `SPAN` kodiert.

Beispiel 1:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
    function width_init()
    {
        ID_Span.style.setExpression("width",
                                    " trueBlueSpan.style.pixelWidth
                                      + oldYellowSpan.style.pixelWidth
                                      ",
                                    "jscript"
                                    );
    }

    function Berechne()
    {alert(ID_Span.style.getExpression("width");}
</SCRIPT>
</HEAD>
<BODY onload= width_init();>
    <SPAN ID="ID_Span"
        STYLE="background-color:lightgreen;
              width:expression( trueBlueSpan.style.pixelWidth
                                + oldYellowSpan.style.pixelWidth
                                )
              ">
    </SPAN>
    <BUTTON onclick= Berechne();>
</BODY>
</HTML>
```

Beispiel 2:

```
ID_Span.style.setExpression("height","document.style.fontSize + 13");
ID_Span.style.setExpression("width","document.body.style.fontSize");
<SPAN ID="ID_Span"
    STYLE="background-color:lightgreen;
          width:expression( trueBlueSpan.style.pixelWidth
                            + oldYellowSpan.style.pixelWidth
                            )
          ">
</SPAN>
```

Beispiel 3 für Sekundenbalken:

```
<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zähler = 0;
```



```

function Init()
{
    // DIV-Eigenschaften festlegen

    // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
    ID_Div1.style.setExpression("width","Zahler *10");

    // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
    ID_Div2.setExpression("innerText","Zahler.toString()");
}

function Uhr()
{
    // Sekunden kumulieren
    Zahler ++;

    // und Anzeige neu berechnen
    document.recalc();
}

function Starten()
{
    if (timerID == null)
    {
        // Start-Button nicht aktivierbar machen
        ID_Button1.disabled = true;

        // Stop-Button aktivierbar machen
        ID_Button2.disabled = false;

        // Uhr neu starten
        timerID = setInterval("Uhr()", 1000);
    }
}

function Stoppen()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
        ID_Button1.disabled = false;
        ID_Button2.disabled = true;
    }
}

function ZurueckSetzen()
{ Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON          ID="ID_Button1"
                    onclick="Starten()"

    >
        Start
    </BUTTON>
    <BR>
    <BUTTON          ID="ID_Button2"
                    DISABLED="true"
                    onclick="Stoppen()"

    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON          ID="ID_Button3"
                    onclick="ZurueckSetzen()"

    >
        Reset
    </BUTTON>
    <BR>

```



```

        <P          STYLE="width:200;color:white;background-color:gray">
            Sekundenbalken
        </P>
    </BODY>
</HTML>

```

Beispiel 4 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl          = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln
    // Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    // ++++++ Routinen der Sekundenzählung
    var SekundenZahler          = 0;
    var SekundenZahlerTimeoutID = null;

    function SekundenZaehlen()    // wird durch Rekursion alle Sekunde neu gestartet
    {
        // Zähler erhöhen
        SekundenZahler++;

        // visuelle Anzeige im Dokument auffrischen, also alle Ausdrücke der Style-Werte
        // neu berechnen und damit alle DIV's neu visualisieren
        document.recalc();
    }

    function SekundenZaehlen_Start()
    {
        // prüfen ob Sekunden zählen nicht bereits läuft
        if (SekundenZahlerTimeoutID == null)
        {
            // nicht aktiv

            // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
            SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
        }
    }

    function SekundenZaehlen_Stop()
    {
        // prüfen ob Sekunden zählen aktiv ist
        if (SekundenZahlerTimeoutID != null)
        {
            // aktiv, also stoppen
            clearInterval(SekundenZahlerTimeoutID);
            SekundenZahlerTimeoutID = null;
        }
    }

    // ++++++ Routine zur Erzeugung Sound-Objekt
    function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
    {
        this.SoundFileUrl          = SoundFileUrl;
        this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
                                                // Timerzeit für Rekursion
        this.SoundFileBeendet      = true;    // kein Sound aktiv
    }

    // ++++++ Routinen zur Wiedergabe Sound-Objekt
    var SoundTimeoutID=0;

    function SoundAbspielen()
    {
        // prüfen ob Sound nicht bereits aktiv ist

```




```

        if (SoundObjekt.SoundFileBeendet)
        {
            // Anzeige initialisieren
            SekundenAnzeigeInit();

            // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
            SekundenZaehlen_Start();

            // Sound erzeugen und sofort starten durch Url-Zuweisung
            ID_BGSound.src=SoundObjekt.SoundFileUrl ;
            SoundObjekt.SoundFileBeendet=false;

            // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
            SoundTimeoutID = setTimeout(
                "SoundAbspielen()",
                SoundObjekt.SoundFileDauerInMillisekundeSekunden
            );
        }
        else
        {
            // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

            // Sound zu Ende
            SoundObjekt.SoundFileBeendet=true;

            // Sekundenzähler stoppen
            SekundenZaehlen_Stop();

            // und Meldung
            var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
            var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
            alert(
                "Wiedergabe beendet\nUngenauigkeit des Timers = "
                + TimerUngenauigkeit1.toString()+ " Sekunden\n"
                + "also " + TimerUngenauigkeit2.toString()+ " Ticks pro Sekunde"
            );
        }
    }

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdruckes.
    //      Für die Neuberechnung des Ausdruckes ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
                                        "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                                + SoundDauerInSekunden.toString()
                                + " Sekunden";
}

```



```

    }

    // ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
    if (ie)
    {
        document.write('<BODY></BODY>');

        document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

        document.write( ' <DIV ID="ID_DIV_Balken"'
            + 'STYLE="background-color:lightblue"'
            + '>'
            + '</DIV>'
            + '<BR>'

        );

        document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
            + 'STYLE="color:hotpink;font-weight:bold"'
            + '>'
            + '</DIV>'
            + '<BR>'

        );

        document.write( ' <DIV ID="ID_DIV_MessLatte"'
            + 'STYLE="color:white;background-color:gray"'
            + '>'
            + '</DIV>'

        );

        // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

        // ---- Sound-Objekt erzeugen anhand globaler Variablen
        SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

        // ---- Sound-Objekt wiedergeben
        SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
    }
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

- .setFullYear()** Jahreszahl vierstellig in lokaler Zeit setzen und optional Monat wie Tag im Monat sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
Script-Objekt Date
- .setHomePage()** Homepage-Lade-Dialogbox aufrufen
- .setHours()** Stunden in lokaler Zeit setzen und optional Minuten, Sekunden, Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
Script-Objekt Date
- .setInterval()** endlos-periodischer Aufruf eines Codes mit jeweiligem vorherigen Warten in Millisekunden (Timer)
(getimte Rekursion)
Der mit setInterval() aufgerufene Code wird zyklisch aktiviert, wobei nach dem ERSTEN Aufruf von setInterval() mit der Folgeanweisung **hinter** setInterval() sofort weitergemacht wird, während die Rekursion parallel läuft. Damit gilt: setInterval() kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da sonst diese Anweisungen während der parallelen rekursiven Ereignisüberwachung bereits abgearbeitet werden.
siehe Objekt window
- Beispiel 1
String window.setInterval("EineFunktion()", 5000);
Zeiger window.setInterval(EineFunktion, 5000); // ohne () kodieren
- Beispiel 2
function callback1(){alert("callback1");}
function callback2(){alert("callback2");}



```

function chooseCallback(Nummer)
{
    switch (Nummer)
    {
        case 0: return callback1;
        case 1: return callback2;
        default: return "";
    }
}
var Nummer = 1;
window.setInterval(chooseCallback(Nummer), 5000);

```

Beispiel 3

```

var SekundenZahler=0;
var timer_id=null;

function ZyklischeAktion()
{
    SekundenZahler++;
    window.status= SekundenZahler + " Sekunden ";

    if ( ( SekundenZahler >= 60 )
        && ( timer_id != null )
    )
    {window.clearInterval(timer_id);}
}

timer_id=window.setInterval(ZyklischeAktion,1000);

```

Beispiel 4

```

var SekundenZahler=0;
var timer_id=window.setInterval("window.status= SekundenZahler++",1000);

```

Beispiel 5:

```

var timerID = null;

function timer()
{
    // tue was
}

function starttimer()
{
    if (timerID == null)
    {timerID = setInterval("timer()", 1000);}
}

function stoptimer()
{
    if (timerID != null)
    {
        clearInterval(timerID);
        timerID = null;
    }
}

```

Beispiel 6 für Sekundenbalken:

```

<HTML>
<HEAD>
<STYLE>
    BUTTON {font-size:14;width:150}
</STYLE>
<SCRIPT>
    var timerID = null;
    var Zahler = 0;

    function Init()
    {
        // DIV-Eigenschaften festlegen

        // DIV-Breite je nach Sekundenanzahl, also dynamische DIV-Breite als Balken
        ID_Div1.style.setExpression("width","Zahler *10");
    }

```



```

        // DIV-Inhalt als Sekundentext, der permanent aktualisiert wird
        ID_Div2.setExpression("innerText", "Zahler.toString()");
    }

    function Uhr()
    {
        // Sekunden kumulieren
        Zahler++;

        // und Anzeige neu berechnen
        document.recalc();
    }

    function Starten()
    {
        if (timerID == null)
        {
            // Start-Button nicht aktivierbar machen
            ID_Button1.disabled = true;

            // Stop-Button aktivierbar machen
            ID_Button2.disabled = false;

            // Uhr neu starten
            timerID = setInterval("Uhr()", 1000);
        }
    }

    function Stoppen()
    {
        if (timerID != null)
        {
            clearInterval(timerID);
            timerID = null;
            ID_Button1.disabled = false;
            ID_Button2.disabled = true;
        }
    }

    function ZurueckSetzen()
    { Zahler = 0; }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <DIV ID="ID_Div1" STYLE="background-color:lightblue" ></DIV>
    <DIV ID="ID_Div1" STYLE="color:hotpink;font-weight:bold"></DIV>
    <BR>
    <BUTTON ID="ID_Button1"
        onclick="Starten()"
    >
        Start
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button2"
        DISABLED="true"
        onclick="Stoppen()"
    >
        Stop
    </BUTTON>
    <BR>
    <BUTTON ID="ID_Button3"
        onclick="ZurueckSetzen()"
    >
        Reset
    </BUTTON>
    <BR>
    <P STYLE="width:200;color:white;background-color:gray">
        Sekundenbalken
    </P>
</BODY>
</HTML>

```

.setMilliseconds()

Millisekunden in lokaler Zeit setzen
Script-Objekt Date



<code>.setMinutes()</code>	Minuten in lokaler Zeit setzen und optional Sekunden, Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setMonth()</code>	Monat in lokaler Zeit setzen und optional Tag im Monat sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setNamedItem()</code>	Attribut hinzufügen anhand Zeiger auf Attribut wenn noch nicht im Feld vorhanden, so Anhängen an das Feldende wenn schon im Feld vorhanden, so überschreiben und Referenz auf das überschriebene Attribut (vor dem Überschreiben) liefern Bsp: Attribut erzeugen <code>document.createAttribute("title");</code> Hinweis: in HTML können Attributnamen groß oder klein geschrieben werden ab IE 6
Beispiel:	<pre> <HTML> <HEAD> <SCRIPT> function Hinzufuegen() { // Attribut mit Wert erzeugen var ZeigerAufAttribut = document.createAttribute("title"); ZeigerAufAttribut.value = "Tooltip-Text "; // Attribut als Feldelement anhängen var ZeigerAufFeld = ID_Div.attributes; ZeigerAufFeld.setNamedItem(ZeigerAufAttribut); } </SCRIPT> </HEAD> <BODY onload="Hinzufuegen();"> <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV> </BODY> </HTML> </pre>
<code>.setSeconds()</code>	Sekunden in lokaler Zeit setzen und optional Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setTime()</code>	Zeitwert in Weltzeit setzen, auch vor 1970 Script-Objekt Date Anzahl der Millisekunden relativ zum 01.01.1970 0 Uhr Weltzeit wenn < 0 so vor obigem Datum
<code>.setTimeout()</code>	einmaliger und somit nicht periodischer Aufruf eines Codes mit genau einem vorherigen Warten in Millisekunden (Timer) Der mit <code>setTimeout()</code> aufgerufene Code wird nicht zyklisch aktiviert, wobei nach dem Aufruf von <code>setTimeout()</code> mit der Folgeanweisung hinter <code>setTimeout()</code> sofort weitergemacht wird. Damit gilt: <code>setTimeout()</code> kann also nicht für Ereignisüberwachung verwendet werden, wenn nachfolgende Anweisungen das Ereignis verarbeiten sollen, da diese Anweisungen bereits während der Ereignisüberwachung abgearbeitet werden. siehe Objekt window

Beispiel 1
`window.setTimeout("alert('Hallo')", 1000);`

Beispiel 2

```

var Text = "Hallo ";
window.setTimeout( "alert("
                    + Text
                    + ")",
                    1000
                );

```

Beispiel 3

```

<SCRIPT>
function Start(ZeigerAufObjekt)
{window.setTimeout("Kode(" + ZeigerAufObjekt.id + ")", 3000);}

function Kode(ID)
{
    var Zeiger = eval(ID);
    Zeiger.style.display="none";
}

```



```

</SCRIPT>
<INPUT TYPE=button VALUE="Unsichtbar in 3 Sekunden"
      ID="ID_Button" onclick=" Start(this)"

```

Beispiel 4

```

var SekundenZahler=0;
var timeout_id=null;

function ZyklischeAktion()
{
    SekundenZahler++;
    window.status= SekundenZahler + " Sekunden ";

    if (      ( SekundenZahler >= 60 )
        && ( timeou_id != null)
        )
    {window.clearTimeout(timeout_id);}

}

timeout_id=window.setTimeout(ZyklischeAktion,1000);

```

Beispiel 5

```

var timeout_id=window.setTimeout("window.status= 'Es ist 1 Sekunde vergangen !'",1000);

```

Beispiel 6 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>

// ++++++ globale Variablen, die verändert werden können
var SoundUrl          = "56sec.mid";
var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

var PixelBreiteProBalkenErweiterung = 10;

// ++++++ Browser-Typ ermitteln
// Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

// ++++++ Routinen der Sekundenzählung
var SekundenZahler          = 0;
var SekundenZahlerTimeoutID = null;

function SekundenZaehlen()    // wird durch Rekursion alle Sekunde neu gestartet
{
    // Zähler erhöhen
    SekundenZahler++;

    // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
    // neu berechnen und damit alle DIV's neu visualisieren
    document.recalc();
}

function SekundenZaehlen_Start()
{
    // prüfen ob Sekundenzählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekundenzählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

```



```

    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl,SoundFileDauerInSekunden)
{
    this.SoundFileUrl                = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
                                           // Timerzeit für Rekursion
    this.SoundFileBeendet              = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(
            "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler          = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdruckes.
    //      Für die Neuberechnung des Ausdruckes ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.

```



```

//          Damit wird der Style-Wert permanent neu berechnet.
//          Damit visualisieren sich die DIV's permanent neu.

// Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
//          also dynamisch anzeigen
ID_DIV_Balken.style.setExpression( "width",
                                   "SekundenZahler * PixelBreiteProBalkenErweiterung"
                                   );

// Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
//          also dynamisch anzeigen
ID_DIV_SekundenZahler.setExpression("innerText", "SekundenZahler.toString()");

// - - - Messlatte statisch anzeigen
ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
ID_DIV_MessLatte.innerText  = "Der Sound dauert "
                               + SoundDauerInSekunden.toString()
                               + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write( ' <BGSOUND ID= "ID_BGSound" LOOP="0">'

    document.write( ' <DIV ID="ID_DIV_Balken"'
                    + 'STYLE="background-color:lightblue"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

                    );

    document.write( ' <DIV ID="ID_DIV_SekundenZahler"'
                    + 'STYLE="color:hotpink;font-weight:bold"'
                    + '>'
                    + '</DIV>'
                    + '<BR>'

                    );

    document.write( ' <DIV ID="ID_DIV_MessLatte"'
                    + 'STYLE="color:white;background-color:gray"'
                    + '>'
                    + '</DIV>'

                    );

    // ++++++ Sound initialisieren und starten mit Laden des Dokumentes

    // ----- Sound-Objekt erzeugen anhand globaler Variablen
    SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

    // ----- Sound-Objekt wiedergeben
    SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
    <!-- BODY-Teil muss leer bleiben!-->
</BODY>
</HTML>

```

.setUTCDate() Tag des Monats in Weltzeit setzen
wenn Tag > Anzahl Tage im Monat, so erfolgt automatisch Monatswechsel und Korrektur aller
Angaben (inklusive Jahr)
Script-Objekt Date

.setUTCFullYear() Jahreszahl vierstellig in Weltzeit setzen und optional Monat wie Tag im Monat
sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
Script-Objekt Date

.setUTCHours() Stunden in Weltzeit setzen und optional Minuten, Sekunden, Millisekunden
sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
Script-Objekt Date



<code>.setUTCMilliseconds()</code>	Millisekunden in Weltzeit setzen Script-Objekt Date
<code>.setUTCMinutes()</code>	Minuten in Weltzeit setzen und optional Sekunden, Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setUTCMonth()</code>	Monat in Weltzeit setzen und optional Tag im Monat sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setUTCSeconds()</code>	Sekunden in Weltzeit setzen und optional Millisekunden sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert Script-Objekt Date
<code>.setYear()</code>	deprecated
<code>.shift()</code>	erstes Feldelement liefern und danach aus dem Feld entfernen Feld hat die Objektklasse JScript-Objekt Array bzw. Script-Objekt Array ab IE 5.5 und NS 6.x siehe auch <code>.pop()</code>
Beispiel:	
<pre>var Feld = new Array(0,1,2,3,4); alert (Feld.shift()); // 0 alert(Feld.join()); // "1,2,3,4"</pre>	
<code>.show()</code>	ein per <code>.createPopup()</code> instanziiertes Popup-Fenster anzeigen Hinweis: Es kann immer nur genau 1 Popup-Fenster angezeigt werden. Das Popup-Fenster wird automatisch geschlossen, wenn ein anderes Objekt den Focus erhält bzw. aktiv wird, z.B. durch Klick des Users außerhalb des Popupfensters ändert Wert der Eigenschaft <code>.isOpen</code> siehe Objekt <code>window.popup</code>

Beispiel für diverse Meldungsfenster per **jeweiligem** PopUp-Fenster (es kann immer nur genau 1 Popup-Fenster angezeigt werden):

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">

var PopUpFensterFeld = new Array();

// nachfolgende Felder beschreiben die PopUp-Fenster und müssen
// identische Anzahl der Feldelemente haben
// Feld-Index ist die Nummer des PopUp-Fensters

var PopUpFenster_RahmenStyle_Feld = new Array
(
    "solid black 4px",
    "solid blue 3px",
    "solid green 2px"
);

var PopUpFenster_HintergrundFarbe_Feld = new Array
(
    "yellow",
    "gray",
    "orange"
);

var PopUpFenster_Inhalt_Feld = new Array
(
    "<B>Inhalt PopUpFenster 0<B>",
    "Inhalt PopUpFenster 1",
    "<TT>Inhalt PopUpFenster 2<TT>"
);

var PopUpFenster_X_Koordinate_Feld = new Array
(
    100,
    150,
    200
```



```

);

var PopUpFenster_Y_Koordinate_Feld = new Array
(
    150,
    200,
    250
);

var PopUpFenster_Breite_Koordinate_Feld = new Array
(
    80,
    140,
    200
);

var PopUpFenster_Hoehe_Koordinate_Feld = new Array
(
    100,
    140,
    180
);

var AnzahlPopUpFenster = PopUpFenster_Hoehe_Koordinate_Feld.length;

function PopupFensterInstanzieren(NummerDesPopUpFensters, ZeigerAufElternFenster)
    // NummerDesPopUpFensters ab 0
{
    PopUpFensterFeld[NummerDesPopUpFensters] =
        ZeigerAufElternFenster.createPopup();
}

function PopupFensterFuellen(NummerDesPopUpFensters)
{
    // Body des Popup-Fensters gestalten
    var PopupFenster_Body =
        PopUpFensterFeld[NummerDesPopUpFensters].document.body;

    PopupFenster_Body.style.backgroundColor =
        PopUpFenster_HintergrundFarbe_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.style.border =
        PopUpFenster_RahmenStyle_Feld[NummerDesPopUpFensters];

    PopupFenster_Body.innerHTML =
        PopUpFenster_Inhalt_Feld[NummerDesPopUpFensters];
}

function PopupFensterAnzeigen(NummerDesPopUpFensters,
    ObjektZuDemPopUpFensterRelativPositioniertIst
)
{
    // Popup-Fenster anzeigen und damit öffnen
    PopUpFensterFeld[NummerDesPopUpFensters].show(
        PopUpFenster_X_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Y_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Breite_Koordinate_Feld[NummerDesPopUpFensters],
        PopUpFenster_Hoehe_Koordinate_Feld[NummerDesPopUpFensters],
        ObjektZuDemPopUpFensterRelativPositioniertIst
    );
}

function PopupFensterSchliessen(NummerDesPopUpFensters)
{
    var Zeiger = PopUpFensterFeld[NummerDesPopUpFensters];

    if (Zeiger.isOpen)
    { Zeiger.hide(); }
}

function Init()
{
    // PopUpFenster instanzieren im aktuellen Fenster (window)

```



```

        for (var i = 0 ; i < AnzahlPopUpFenster; i++)
        {
            PopupFensterInstanzieren(i, window);
            PopupFensterFuellen(i);
        }
    }
</SCRIPT>
</HEAD>
<BODY onload="Init();">
    Durch Klick ausserhalb des Popup-Fensters wird dieses auch automatisch geschlossen.
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 anzeigen"
        onclick=" PopupFensterAnzeigen(0, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 anzeigen"
        onclick=" PopupFensterAnzeigen(1, document.body);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 anzeigen"
        onclick=" PopupFensterAnzeigen(2, document.body);"
    >
    <BR>
    <INPUT TYPE=button
        VALUE="PopUpFenster 0 schliessen"
        onclick=" PopupFensterSchliessen(0);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 1 schliessen"
        onclick=" PopupFensterSchliessen(1);"
    >
    <INPUT TYPE=button
        VALUE="PopUpFenster 2 schliessen"
        onclick=" PopupFensterSchliessen(2);"
    >
</BODY>
</HTML>

```

`.ShowBrowserUI()` öffnen einer IE-Dialogbox (UI = User-Interface) bezüglich
 Spracheinstellungen
 oder Favoritenverwaltung
 oder Sicherheitseinstellungen
 Inwieweit eine **geöffnete** Dialog-Box dann per Windows Script Host z.B. per VBScript programmierbar ist,
 wird hier nicht betrachtet.
 Ein User, dem diese Boxen durch eine Webseite geöffnet werden, sollte stets misstrauisch ein.
 siehe Objekt `window.external`

Beispiel:

```
<BUTTON onclick="window.external.ShowBrowserUI('LanguageDialog', null)">Spracheinstellungen</BUTTON>
```

`.showHelp()` Helpdatei anzeigen (*.chm und *.htm)
 siehe Objekt `window`

`.showModalDialog()` Modales Dialog-Fenster erzeugen und anzeigen (modaler Dialog) sowie automatisch focussieren
 siehe Objekt `window`

Beispiel 1:

```

<SCRIPT>
function ErmittleZufallsZahl(Faktor)
{return parseInt(Math.random() * Faktor);}

function BoxHoeheErmittleIn()
{
    var OptionsFeld = ID_Formular.ID_Select.options;
    var OptionsFeldIndex = ID_Formular.ID_Select.selectedIndex;
    var OptionsWert = OptionsFeld [OptionsFeldIndex].text;

    // auf Auswahl " Zufallshoehe" prüfen
    if (OptionsWert.indexOf("Zufallshoehe") > -1 )
    {OptionsWert = ErmittleZufallsZahl(document.body.clientHeight);}

    // Features für Boxöffnen ermitteln
    var BoxHoeheFeatures ="dialogHeight:" + OptionsWert + "px;";

    // und liefern

```



```

        return BoxHoeheFeatures;
    }

    function BoxOeffnen()
    {
        var BoxHoehe Features= BoxHoeheErmitteln();

        window.showModalDialog( "test.htm",
                                "",
                                BoxHoeheFeatures
                                );
    }
</SCRIPT>
<FORM NAME="ID_Formular">
    wachle Boxhoehe in Pixel
    <SELECT NAME="ID_Select">
        <OPTION>Zufallshoehe
        <OPTION>150
        <OPTION>200
        <OPTION>250
        <OPTION>300
    </SELECT>
    oeffne Modal Dialog Box
    <INPUT TYPE="button" VALUE="Box oeffnen" onclick="BoxOeffnen()">
</FORM>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeige()
    {
        // Zeiger auf die Formularelemente holen
        var FormularElemente = ID_Formular.elements;

        // Formulardaten in einem privaten Objekt kapseln als Argument für modalen Dialog
        var FormularDaten = new Object();
        FormularDaten.firstName = FormularElemente.ID_Input1.value;
        FormularDaten.lastName = FormularElemente.ID_Input2.value;

        // Fenster als modalen Dialog anzeigen und dabei Eigenschaft .dialogArguments füllen
        // Mit der Übergabe der Daten erfolgt das öffnen des neuen Fenster, das
        // sich auf die namensgleichen Eigenschaften von FormularDaten
        // beruft, deren Bezeichner mit in den Argumenten übergeben werden
        window.showModalDialog( "test.htm",
                                FormularDaten,
                                "dialogHeight:300px; dialogLeft:200px;"
                                );
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM ID= "ID_Formular">
        Vorname:
        <INPUT TYPE="text" NAME="ID_Input1" VALUE="Vorname">
        <BR>
        Nachname:
        <INPUT TYPE="text" NAME="ID_Input2" VALUE="Nachname">
    </FORM>
    <BR>
    <BUTTON onclick="Anzeige();" >Formulardaten im modalen Dialog anzeigen</BUTTON>
</BODY>
</HTML>

```

test.htm enthält:

```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeigen()
    {
        var DialogArgumente = window.dialogArguments;

        // Es müssen namensidentische Bezeichner der Eigenschaften von

```



```

//      DialogArgumente verwendet werden:
//      firstName und lastName werden in der
//      aufrufenden Webseite definiert
//      und müssen hier ebenfalls verwendet werden
document.writeln("Vorname = " + DialogArgumente.firstName);
document.write("Nachname = " + DialogArgumente.lastName);
}
</SCRIPT>
</HEAD>
<BODY onload="Anzeigen();">
</BODY>
</HTML>

```

.showModelessDialog() nicht-modales Fenster erzeugen, aber anzeigen nur, wenn Focus geändert wird
 verwendbar für Desing von Menüs
 Tooltips
 siehe Objekt window

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
var Vorname=""; // muss global sein da in Test.htm benutzt

function VornameAnzeigen()
{
    showModelessDialog( "Test.htm", // Url der Dialog-Box
                        window,
                        "status:false;dialogWidth:300px;dialogHeight:300px"
                      );
}

function VornameAktualisieren() // Funktion wird in Test.htm aufgerufen
{ ID_Span.innerText = Vorname; }

</SCRIPT>
</HEAD>
<BODY>
<P> aktueller Vorname ist :
    <SPAN ID="ID_Span"
        STYLE="color:red;font-size:24">
        unbekannt
    </SPAN>
</P>
<INPUT TYPE="button"
        VALUE="öffnen der Modeless Dialog Box"
        onclick="VornameAnzeigen()">

</BODY>
</HTML>

```

Kode für *Test.htm*, also dem Inhalt der Box

```

<HTML>
<HEAD>
<TITLE>Test.htm</TITLE>
<SCRIPT>
function VornameAktuellAnzeigen()
{
    var DialogArgumente = dialogArguments;
    DialogArgumente.Vorname = ID_Input.value;
    DialogArgumente.VornameAktualisieren();
    // Aufruf einer Funktion aus
    // Elterndokument
}

function Init()
{
    var DialogArgumente = dialogArguments;
    DialogArgumente.Vorname = "unbekannt";
    DialogArgumente.VornameAktualisieren();
    // Aufruf einer Funktion aus
    // Elterndokument
}

</SCRIPT>
</HEAD>
<BODY>

```



```

<LABEL FOR="ID_Input" ACCESSKEY="v">
    Gib den
    <SPAN STYLE="text-decoration:underline">
        V
    </SPAN>ornamen ein :
</LABEL>
<INPUT ID= "ID_Input">
<BR><BR>
<INPUT VALUE="Anzeige aktueller Vorname und Box offen lassen"
    TYPE=button
    onclick=" VornameAktuellAnzeigen();">

<INPUT VALUE=" Anzeige aktueller Vorname und Box schliessen"
    TYPE=button
    onclick=" VornameAktuellAnzeigen();window.close();">

<INPUT VALUE="Init"
    TYPE=button
    onclick=" Init();window.close();"
>
</BODY>
</HTML>

```

.simpleTimeToSegmentTime() Wert der Simple-Timeline eines Elementes in den korrespondierenden Wert der Segment-Timeline des Elementes konvertieren
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    window.onload = Rekursion;    // ohne () kodieren

    function Rekursion()
    { window.setInterval(Anzeige, 100);}

    function Anzeige()
    {
        ID_Span2.innerHTML = " simpleTimeToSegmentTime: "
            + (ID_Animation. simpleTimeToSegmentTime (
                ID_div.currTimeState.activeTime
            )
        );
        ID_Span3.innerHTML = " segmentTime: "
            + (ID_Animation.currTimeState.segmentTime);
    }
</SCRIPT>
</HEAD>
<BODY>
<SPAN ID="ID_Span1"
    CLASS="time_line_klasse"
    DUR="1"
    REPEATCOUNT="indefinite"
    onrepeat="innerText=parseInt(document.body.currTimeState.activeTime);"
>
    0
</SPAN>
<DIV ID="ID_Div"
    CLASS="time_line_klasse"
    STYLE="position:absolute; left:50px; top:250px; width:100px; height:100px;"
>
</DIV>
<t:ANIMATEMOTION ID="ID_Animation"
    TARGETELEMENT="ID_div"
    TO="250,0"
    DUR="3"
    AUTOREVERSE="true"
>
</t:ANIMATEMOTION>
<SPAN ID="ID_Span2">
    simpleTimeToSegmentTime:

```



```

</SPAN>
<BR>
<SPAN ID="ID_Span3">
    segmentTime:
</SPAN>
</BODY>
</HTML>

```

`.sin()` Sinus im Bogenmass
liefert Wert von -1 bis +1
siehe Script-Objekt Math

`.Skip()` in der offenen Datei die nächsten Zeichen überlesen (egal ob newline dabei ist oder nicht)
verändert den Satzzeiger

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.Skip(1);
alert(DateiOffen.Read(1));
DateiOffen.Close();

```

`.SkipLine()` in der offenen Datei die nächsten Zeilen überlesen
verändert den Satzzeiger
sinnvoll nur verwendbar, wenn mindestens 1 newline-Zeichen in der Datei auftaucht

Beispiel:

```

var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("Test");
DateiOffen.Write("123456789");
DateiOffen.Close();
DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 1,false,0);
DateiOffen.SkipLine(1);
alert(DateiOffen.ReadLine());
DateiOffen.Close();

```

`.slice()` Elementefolge aus Quellfeld als neues Feld (Zielfeld) liefern
nachträgliche Änderung der Elementanzahl im Quellfelde wirken sich nicht auf Zielfeld aus
(keine dynamische Verkettung)
wenn Quellfeld **nicht** String **und nicht** Number enthält:
neues Feld hat Feldelemente als Zeiger auf die Elemente also nicht als Wertkopie
Werte-Änderung im Quellfeld bewirkt sofortige Änderung im Zielfeld, da identische Zeiger
vorliegen
wenn Quellfeld Strings oder Number enthält:
neues Feld hat Feldelemente als Wertkopien aus dem Quellfeld
Werte-Änderung im Quellfeld bewirkt keine Änderung im Zielfeld, da nicht identische Zeiger
vorliegen
Felder haben die Objektklasse Script-Objekt Array

Beispiel:

```

var Feld1 = new Array(0,1,2,3,4);
var Feld2 = Feld1.slice(0,1);
alert(Feld2.join()); // "0"
var Feld3 = Feld1.slice(1);
alert(Feld3.join()); // "1,2,3,4"

```

Beispiel:

```

var QuellFeld=new Array("a","b","c")
var ZielFeld=QuellFeld.slice(0,2) // Zielfeld enthält Wertkopien also ["a","b"]

```

Beispiel:

```

var Variable1="Hallo ";
var Variable2="Du";
var Variable3="!";

var QuellFeld=new Array(Variable1,Variable2,Variable3);
var ZielFeld=QuellFeld.slice(0,-1) ; // 2 verwendet, denn Länge 3 minus 1 ist 2
// Zielfeld enthält Wertkopien also [ Zeiger_Auf_Variable1,
// Zeiger_Auf_Variable2
// ]

```

`.slice()` Teilkette aus String oder Stringliteral liefern als neuen String
siehe Script-Objekt String



Beispiel:

```
var StringLiteral = "StringLiteral";
StringLiteral.length      liefert 13
StringLiteral.slice(3,-5) liefert "ingLit" // 13 + (-5) ergibt 8
StringLiteral.slice(3,5)  liefert "ing"
"StringLiteral".slice(3,5) liefert "ing"
```

`.small()` HTML-Tag <SMALL> erzeugen in der Form <SMALL>text</SMALL>
siehe Script-Objekt String

Beispiel:

```
var StringLiteral = "Text der SMALL wird"
var HTML_Kette = StringLiteral.small();
HTML_Kette = "Text der SMALL wird".small();
```

entspricht <SMALL>Text der SMALL wird</SMALL>

`eval(HTML_Kette);` erzeugt Fehler, da HTML-Code von `eval` nicht ausgeführt werden kann
`document.write(HTML_Kette);`

`.sort()` Feldelemente physisch sortieren
sind zwei direkt zusammenliegende Feldelemente wertmäßig identisch, so werden deren Positionen im Feld nicht geändert
ein Feldelement mit Wert `undefined` landet am Feldende:
Liegen mehrere Feldelemente mit Wert `undefined` vor, so landen diese in der Reihenfolge vor der Sortierung am Ende des Feldes nach der Sortierung
Standardsortierung : laut ASCII-Zeichensatz und aufsteigend
Feld hat die Objektklasse Script-Objekt Array

Beispiel:

```
var Feld1 = new Array("4","3","2","1","0");
Feld1.sort(); // Feld1 ist ungültig
alert(Feld1.join()); // "0,1,2,3,4"
```

Beispiel für numerische Feldelemente:

```
function Sortiere(Arg1, Arg2)
{return Arg1 - Arg2;} // Arg1 > Arg2 so positiver Wert
// Arg1 < Arg2 so negativer Wert
// Arg1 == Arg2 so 0

zeiger_auf_feld.sort(Sortiere);
```

`.splice()` Feldelemente-Folge aus dem Feld entfernen ab Indexposition
optional neue Objekte in das Feld einfügen ab Indexposition
alle entfernte Feldelemente in einem neuen Feld liefern in der Reihenfolge des Entfernens
Felder haben die Objektklasse Script-Objekt Array

Beispiel:

```
var Feld1 = new Array(0,0,1,2,3,4);
var Feld2 = Feld1.splice(0,1);
alert(Feld2.join()); // "0,1,2,3,4"
var Feld3 = Feld1.splice(0,2,-1,0);
alert(Feld3.join()); // "-1,0,1,2,3,4"
```

`.split()` String bzw. Stringliteral zerlegen und als Feld der Teilketten liefern
Zerlegung von links nach rechts
Feldelemente-Folge wie Zerlegungsfolge
siehe Script-Objekt String

Beispiel:

```
var StringLiteral = "StringLiteral";
var Feld = StringLiteral.split("r", 2); // Feld enthält 2 Elemente mit "St" und "ingLiteral"
Feld = StringLiteral.split("r", 1); // Feld enthält 1 Elemente mit "St"
Feld = StringLiteral.split("r"); // Feld enthält 3 Elemente mit "St" und "ingLite" und "al"
Feld = "StringLiteral".split("r"); // Feld enthält 3 Elemente mit "St" und "ingLite" und "al"
```

`.sqrt()` Quadratwurzel ziehen
siehe Script-Objekt Math

`.start()` Start der Scrollaktion eines marquee Objektes
Anzahl der Scrollaktionen laut Eigenschaft `.loop`
erzeugt **nicht** das Ereignis `onstart`
Hinweis: Eigenschaften `.scrollLeft` und `.scrollTop` sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.

`.startDownload()` Start des Download

Beispiel für Funktion:




```
function Anzeige( Kette )
{
    var Feld = Kette.split("\n");
    var FeldLaenge = Feld.length;

    if (FeldLaenge > 0)
    {
        for ( var Index=0; Index < FeldLaenge; Index++)
        {alert("Index = " + Index + ' Teil = ' + Feld[Index]);}
    }
}
```

Beispiel:

```
<HTML XMLNS:IE>
<HEAD>
<SCRIPT>
    function NachDemDownload(DownLoadedFileContent)
    {alert (DownLoadedFileContent);}
</SCRIPT>
</HEAD>
<BODY>
<IE:Download ID="ID_IETag" STYLE="behavior:url(#default#download)" >
<P>
Click
<A HREF="javascript:ID_IETag.startDownload('download.htm', NachDemDownload)">
hier
</A>
zum Start des Downloads dieser Seite.
</BODY>
</HTML>
```

.stop() Stopp der aktuellen und laut Eigenschaft .loop noch offener Scrollaktionen eines marquee Objektes erzeugt **nicht** das Ereignis onstop
Hinweis: Eigenschaften .scrollLeft und .scrollTop sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.

.stop() aktive Wiedergabe stoppen, also beenden
verändert Eigenschaft .playState
Es kann zu jedem Zeitpunkt nur genau 1 Media-Datei wiedergegeben werden, es sei denn, man öffnet mehrere Browser-Instanzen.
Die Wiedergabe ist nicht möglich, wenn der User in der Media Bar per Links navigiert, da damit die

aktuelle Url in der Media Bar geändert wird gegenüber der Url der gerade wiedergegebenen Media-Datei.
Eine aktive Wiedergabe wird durch HTML-Navigation sofort gestoppt.
Achtung: Nur wenn eine Media-Datei gerade wiedergegeben wird, dann sind per Script folgende Methoden nutzbar:
.getItemInfo()
.getAttributeName()
Eigenschaften nutzbar:
.attributeCount
Es ist also vorher die Eigenschaft .playState auf Werte > 10 zu prüfen.
Achtung: Wird die Methode .stop() innerhalb von 10 Sekunden nach dem ERSTEN Aufruf erneut aufgerufen, dann wird eine Standard-Seite in das Medien-Fenster geladen.
siehe Methoden .playURL() .playNext() und Eigenschaft .playState
siehe Behavior .style.mediaBar

.strike() HTML-Tag <STRIKE> erzeugen in der Form <STRIKE>text</STRIKE>
siehe Script-Objekt String

Beispiel:

```
var StringLiteral    ="Text der STRIKE wird"
var HTML_Kette      = StringLiteral.strike();
HTML_Kette           = "Text der STRIKE wird".strike();
```

entspricht <STRIKE>Text der STRIKE wird</STRIKE>

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

String() konvertiert eine Instanz zu String
ist identisch mit Methode .toString()

.sub() HTML-Tag <SUB> erzeugen in der Form _{text}
siehe Script-Objekt String



Beispiel:

```
var StringLiteral    ="Text der SUB wird"
var HTML_Kette      =StringLiteral.sub();
HTML_Kette          ="Text der SUB wird".sub();
```

entspricht _{Text der SUB wird}

```
eval(HTML_Kette);           erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);
```

.submit() löst für Formular das Event onsubmit aus, dessen Eventhandler die Submit-Aktion beinhaltet, die gestartet wird VOR dem Senden der Elemente (Browser sendet)

Eventhandler muss liefern: return true, für Ausführung des Submit durch Browser
 return false; für Nicht-Ausführung des Submit durch Browser

Beispiel: Man beachte, dass NAME-Attribut für alle zu sendenden bzw. rückzusetzenden Elemente **Pflicht** ist !!

```
<HTML>
<HEAD>
<SCRIPT>
function EventHandler_AktionVorSubmit()
{
    // ein Hinweis im Textarea anzeigen
    ID_Textarea.value += "Formular senden ????";}

    // wirklich senden ???
    return( confirm("Wirklich senden ?"));
    // true so Submit durch Browser ausführen lassen
    // false so kein Submit durch Browser
}
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <FORM NAME="Formular" ACTION=" ..." METHOD=" "
        onsubmit="EventHandler_AktionVorSubmit();">
        <INPUT TYPE="text" NAME="InputText" VALUE="Text eingeben">
        <BR>
        <BUTTON onclick="form.submit();">OnSubmit auslösen</BUTTON>
        <BR><BR>
        <INPUT TYPE="submit" VALUE="oder hiermit senden"
            onclick="form.submit()"
        >
    </FORM>
</DIV>
<TEXTAREA ID="ID_Textarea" VALUE="" ROWS="5" COLS="75"></TEXTAREA>
</BODY>
</HTML>
```

.substr() Teilkette aus String oder Stringliteral liefern als neuen String
siehe Script-Objekt String

.substring() Teilkette aus String oder Stringliteral liefern als neuen String
siehe Script-Objekt String

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.substring(3,3) liefert ""
StringLiteral.substring(3,5) liefert "in"
"StringLiteral".substring(3,5) liefert "in"
```

Beispiel für Erzeugung von Vornull mit Vorzeichen bei Ziffern-Zeichenkette, die numerisch <1 bzw. > -1 ist

```
function vornull(zeichenkette)
{
    var funktionswert=zeichenkette;

    if (    (zeichenkette.substring(0, 1) == ",")
        ||  (zeichenkette.substring(0, 1) == ".")
        )
    {funktionswert = "0" + zeichenkette }

    if (    (zeichenkette.substring(0, 2) == "-.")
        ||  (zeichenkette.substring(0, 2) == "-.")
        )
    {funktionswert = "0" + zeichenkette }
```



```

    { funktionswert = "-0" + zeichenkette.substring(1)}

    return funktionswert;
}

```

.substringData() Teilkette aus einem Objekt lesen

.sup() HTML-Tag <SUP> erzeugen in der Form ^{text}
siehe Script-Objekt String

Beispiel:

```

var StringLiteral    ="Text der SUP wird"
var HTML_Kette       = StringLiteral.sup();
HTML_Kette           = "Text der SUP wird".sup();

```

entspricht ^{Text der SUB wird}

eval(HTML_Kette); erzeugt Fehler, da HTML-Code von eval nicht ausgeführt werden kann
document.write(HTML_Kette);

.swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
nur sichtbar wenn Endetag geparkt
DOM wird geändert

Beispiel:

```

<SCRIPT>
function Tauschen()
{Liste.children(0).swapNode(Liste.children(1)); }
</SCRIPT>
<UL ID = Liste>
<LI>Listeneintrag 1
<LI>Listeneintrag 2
<LI>Listeneintrag 3
<LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Tauschen" onclick = "Tauschen()">

```

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern
siehe tags Collection des DOM

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
var ZeigerAufFeldAllerPTag = document.all.tags("P");
if (ZeigerAufFeldAllerPTag!=null)
{
    for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
    { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
}
</SCRIPT>

```

.taintEnabled() Data-Tainting (Daten verwerfen) per navigator Objekt
ab IE 6.x

.taintEnabled() Data Tainting-Verfügbarkeit
ab IE 6.x
siehe Objekt window.clientInformation

.tan() Tangens im Bogenmass
liefert Wert von -1 bis +1
siehe Script-Objekt Math

.toDateString() kompletten Inhalt des Date-Objektes als String liefern
Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
Script-Objekt Date

.toExponential() String liefern, der den Wert in Exponential-Darstellung enthält
auch bei numerischem Literal z.B. "10.2345678e+13"
IE ab 5.5, NS ab 6.x
siehe Script-Objekt Number

Beispiel:

```

var num=77.1234
num.toExponential()    liefert "7.71234e+1"
num.toExponential(4)) liefert "7.7123e+1"
num.toExponential(2)) liefert "7.71e+1"
77.1234.toExponential() liefert "7.71234e+1"
77 .toExponential())  liefert "7.7e+1"

```

.toFixed() String liefern, der den Wert in Festkomma-Darstellung enthält



auch bei numerischem Literal z.B. "10.2345678"
 IE ab 5.5, NS ab 6.x
 siehe Script-Objekt Number

Beispiele:

```
var num=10.1234
num.toFixed()      liefert "10"
num.toFixed(4)     liefert "10.1234"
num.toFixed(2)     liefert "10.12"
0.124.toFixed(2)   liefert "0.12"
0.125.toFixed(2)   liefert "0.13"
0.126.toFixed(2)   liefert "0.13"
0.045.toFixed(2)   liefert "0.05"
1234.56789.toFixed(4) liefert "1234.5679"
```

.toGMTString() deprecated

.toLocaleDateString() kompletten Inhalt des Date-Objektes als String in lokaler Einstellung des Betriebssystems auf User-PC liefern
 immer verwenden für Jahre von 1601 bis 1999 --> siehe **.toLocaleString()**
 Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
 Script-Objekt Date

.toLocaleLowerCase() String bzw. Stringliteral nach neuen String kopieren und dort nach Kleinbuchstaben umwandeln
 laut aktuelle Sprach-Einstellungen der Umgebung auf PC des Users
 nur IE ab 5.5
 siehe Script-Objekt String

.toLocaleString() Wert eines Objektes in System-lokale Einstellungen umwandeln
 System-lokale Einstellung z.B. Ländereinstellung, Uhrzeitformat
 Konvertierung: Analog wie z.B. bei Excel, das die lokalen Einstellungen ausliest.
 nur anwenden für Anzeige des konvertierten Wertes:
 Berechnungen mit dem Wert immer unkonvertiert vollziehen, da sämtliche
 Berechnungsfunktionen **nur** das interne, also unkonvertierte
 Format kennen (**nicht** analog zu Excel)
 Wenn das Objekt ein Script-Objekt Array ist, also Feldelemente in lokale Einstellungen konvertiert werden
 sollen, dann wird ein String geliefert, der die Feldelemente in der Reihenfolge im Feld
 und diese getrennt durch **dasjenige** Trenner-Zeichen laut **lokale** Einstellungen enthält.
 Wenn das Objekt eine Script-Objekt Date ist, so wird der Wert von dem Objekt in den lokalen
 Datumeinstellungen geliefert. Dabei sind nur Jahreszahlen von 1601 bis 9999 zulässig.
 Andere Jahresangaben werden nicht konvertiert.
 Wenn das Objekt ein Script-Objekt Number ist, so werden die lokalen Einstellungen zum Zahlenformat
 geliefert.
 Wenn das Objekt ein Script-Objekt Object ist, so werden nur dann lokale Einstellungen berücksichtigt,
 insoweit das Objekt diese berücksichtigen kann. Ein String wird immer geliefert.

Beispiel für Konvertierung eines Feld mit Gleitkomma-Werten:

```
var Feld = new Array(6);
var Wert = 3,201,300.20; // in JScript ist das Dezimalkomma der Punkt
                        // der Tausendertrenner das Komma

// Feld füllen
for( var i = 0; i < 7; i++)
{
    Wert +1 = 10;
    Feld [i] = Wert;
}

alert(Feld.toLocaleString());
```

Beispiel für Konvertierung eines Datums:

```
var Jetzt = new Date();
alert(Jetzt.toLocaleString());
```

.toLocaleString() kompletten Inhalt des Date-Objektes als String in lokaler Einstellung auf User-PC liefern
 nur für Jahre ab 2000
 aber für Jahre von 1601 bis 1999 immer laut lokale Einstellungen des Betriebssystems auf User-PC
 siehe **.toLocaleDateString()**
 Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
 Script-Objekt Date

.toLocaleTimeString() Zeitwert des Date-Objektes als String in lokaler Einstellung des Betriebssystems auf User-PC liefern



Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
Script-Objekt Date

.toLocaleUpperCase() String bzw. Stringliteral nach neuen String kopieren und dort nach Grossbuchstaben umwandeln
laut aktuelle Sprach-Einstellungen der Umgebung auf PC des Users
nur IE ab 5.5
siehe Script-Objekt String

.toLowerCase() String bzw. Stringliteral nach neuen String kopieren und dort nach Kleinbuchstaben umwandeln
siehe Script-Objekt String

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.toLowerCase()      liefert "stringliteral"
"StringLiteral".toLowerCase()    liefert "stringliteral"
```

.toFixed() liefert String, der die Nachkommastellen enthält
auch bei numerischem Literal z.B. "10.2345678"
IE ab 5.5, NS ab 6.x
siehe Script-Objekt Number

Beispiele:

```
var num=5.123456
num.toFixed()           liefert "5.123456"
num.toFixed(4)          liefert "5.123"
num.toFixed(2)          liefert "5.1"
num.toFixed(1)          liefert "5"

1250 .toFixed(2)        liefert "1.3e+3"
1250 .toFixed(5)        liefert "1250.0"
1234.56789.toFixed(2)   liefert "1.2e+3"
1234.56789.toFixed(9)   liefert "1.234.56789" entspricht also e+0
```

.toString() Wert eines Objektes in einen String umwandeln
wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma getrennt geliefert
wenn Objekt ein Script-Objekt Boolean ist, dann "true" bzw "false" geliefert (Kleinschreibung !)
wenn Objekt ein Script-Objekt Error ist, dann die Fehlermeldung geliefert (Objekt-Wert ist der Fehlercode)
wenn Objekt ein Script-Objekt Function ist, dann Quellcode der Funktion geliefert (inklusive Funktionskopf)
wenn Objekt ein Script-Objekt Object ist, so wird geliefert:
"["objectname"]"
mit objectname als konkreter Bezeichner der Objektklasse bzw. des Objektes fettgedrucktes wird so geliefert wie angegeben
wenn der Bezug vor .toString() ein Wert laut ID-Attribut oder NAME-Attribut ist, dann wird die Objektklasse geliefert

Beispiel 1:

```
var Wert = 33,33;
alert(Wert.toString(2) + ' ' + Wert.toString(16) + ' ' + Wert.toString(10));
```

Beispiel 2:

```
<BUTTON ID="ID_Button" .... > .... </BUTTON>
ID_Button.toString() liefert "button"
```

Beispiel 3: Numerischen Wert als Zeichenkette liefern und dabei Dezimalpunkt zu Dezimalkomma umwandeln

Es wird angenommen, dass genau ein Dezimalpunkt vorkommt.

```
function punkt_zu_komma (numerischer_wert)
{
    var zeichenkette = numerischer_wert.toString();
    var funktionswert=zeichenkette;

    var pos_punkt = zeichenkette.indexOf(".");

    if(pos_punkt >=0)
    {
        funktionswert =    zeichenkette.substring(0, pos_punkt)
                        + ","
                        + zeichenkette.substring(pos_punkt + 1, zeichenkette.length);
    }

    return zeichenkette;
}
```

.getTimeString() Zeitwert des Date-Objektes als String liefern



Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
Script-Objekt Date

.toUpperCase() String bzw. Stringliteral nach neuen String kopieren und dort nach Grossbuchstaben umwandeln
siehe Script-Objekt String

Beispiel:

```
var StringLiteral ="StringLiteral";
StringLiteral.toUpperCase() liefert "STRINGLITERAL"
"StringLiteral".toUpperCase() liefert "STRINGLITERAL"
```

.toUTCString() Zeitwert des Date-Objektes als String in Weltzeit liefern
Achtung: nur für Anzeige etc. verwenden und nicht für Berechnungen !
Script-Objekt Date

unescape ab Javascript 1.5 (Netscape 6.x) deprecated und zu ersetzen durch die Methoden
decodeURI() und decodeURIComponent()
dekodiert einen mit der Methode escape() kodierten String (siehe escape())

.unshift() Werte dem Feld voransetzen (auch wenn Feld leer ist)
Feld hat die Objektklasse JScript-Objekt Array bzw. Script-Objekt Array
ab IE 5.5 und NS 6.x

Beispiel:

```
var Feld = new Array(0,1,2,3,4);
alert (Feld.unshift(-1));
alert(Feld.join()); // "-1,0,1,2,3,4"
```

url() Diese Methode ist nicht direkt als Methode des Style-Objektes implementiert und darf daher
nicht mit Punktnotation kodiert werden.
Diese Methode wird nur in Verbindung mit dem Style Objekt verwendet.

Beispiele für Bild:

```
<STYLE>
    .style1 { background:beige url(sphere.jpg) no-repeat top center }
    .style2 { background:ivory url(sphere.jpeg) no-repeat bottom right }
</STYLE>

<SPAN onclick="this.style.background='beige url(sphere.jpeg) no-repeat top center'"></SPAN>

<STYLE>
    .setUrl { background-image: url(sphere.jpg) }
    .loseUrl { background-image: url(none) }
</STYLE>
<SPAN STYLE="font-size:14"
    onmouseover="this.className='setUrl'"
    onmouseout="this.className='loseUrl'"
>
</SPAN>
```

Beispiele für Behavior:

```
url(#objID) mit objID als ID des OBJECT-Tags
url(#default#behaviorName) eines Standard-Behaviors des IE

STYLE="behavior:url(a1.htc) url(a2.htc)"

<STYLE>
    .Klasse { behavior:url(#myObject) }
</STYLE>

<STYLE>
    DIV { behavior:url(fly.htc) url (zoom.htc) url (fade.htc)}
</STYLE>
```

Beispiel für Cursorformen aus **Datei** und **nicht** für im Browser implementierte Standard-Cursorformen:
url('mycursor.cur')

Hinweise: alert (zeiger_auf_objekt.style.cursor); liefert beim IE den String 'url(cursor_form)'
mit cursor_form für den aktuellen **und** gesetzten Cursor.

style.cursor ist standardgemäß mit einer Leerkette belegt, solange **kein** Cursor gesetzt wird
kann nicht mit url(cursor_form) belegt werden, wenn es sich um eine **standardgemäß im**
Browser implementierte Cursorform handelt wie z.B. 'hand' oder 'normal'.
Grund: Diese Cursorformen sind keine Dateien, obwohl alert den String



'url(cursor_form)' anzeigt.

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
  var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
  var Text = "";

  if (ZeigerAufFeldAllerURN1 != null)
  {
    for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
    {Text += ZeigerAufFeldAllerURN1.item(i).id + ' ';}
    alert (Text);
  }
</SCRIPT>
```

.UTC() Inhalt des Date-Objektes setzen nach Weltzeit
(auch wenn Objekt bereits initialisiert wurde per new Anweisung)
sollte ein Wert außerhalb des Wertebereiches sein, so wird automatisch korrigiert
Script-Objekt Date

.valueOf() Wert eines JScript-Objektes bzw. Instanz eines JScript-Objektes ermitteln
nicht bei Script-Objekt Math und Script-Objekt Error (auch nicht bei Instanzen dieser Objekte)
Wert ist im Datentyp des Objektes, aber:
wenn Objekt ein Script-Objekt Array ist, dann Elemente in der Feldreihenfolge und mit Komma
getrennt geliefert (identisch in der Wirkung mit .toString() und der Array-Methode
.join())
wenn Objekt ein Script-Objekt Boolean ist, dann true bzw false geliefert (kein String !)
wenn Objekt ein Script-Objekt Date ist, dann Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr
geliefert
wenn Objekt ein Script-Objekt Function ist, dann Zeiger auf Funktion geliefert
wenn Objekt ein Script-Objekt Number ist, dann numerischen Wert geliefert
wenn Objekt ein Script-Objekt Object ist, so Zeiger geliefert
siehe auch .toLocaleString() und .toString()

.write() Text bzw. HTML-Ausdruck in das Dokument ausgeben und anzeigen bzw. sofort parsen

Hinweis:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen** HTML-Dokumentes, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

Plain-Text
HTML-Text
Script

Beispiel für Ersatz von document.write() durch die Eigenschaft .innerHTML (nur IE):

```
<SCRIPT>
  var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

  function Laden()
  {
    ID_Div.innerHTML=Kette;           // wird sofort geparkt, also IMG-Tag ausgeführt
                                     // ( anstelle von eval()
                                     // bzw. document.write() )

    ID_IMG.SRC="";
    ID_IMG.style.display="block";

    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
  }

  function IMGAltTextAufFehlerMeldung ()
  {
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
  }
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>
```

.Write() in die offenen Datei zeichenweise schreiben (ab Position laut Satzzeiger)



verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.Close();
```

.WriteBlankLines() in die offenen Datei newline-Zeichen schreiben (ab Position laut Satzzeiger)
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.Write("123456789");
DateiOffen.WriteBlankLines(1);
DateiOffen.Close();
```

.WriteLine() in die offenen Datei genau 1 Zeile schreiben (ab Position laut Satzzeiger)
wobei am Ende automatisch genau ein newline-Zeichen geschrieben wird
verändert den Satzzeiger

Beispiel:

```
var DateiSystem = new ActiveXObject("Scripting.FileSystemObject");
var DateiOffen = DateiSystem.OpenTextFile("c:\\test.txt", 2,true,0);
DateiOffen.WriteLine("123456789");
DateiOffen.Close();
```

.writeln() Text bzw. HTML-Ausdruck in das Dokument ausgeben und anzeigen bzw. sofort parsen

Hinweis:

Die **ERSTE** Erzeugung eines HTML-Tags bewirkt das **automatische Öffnen eines neuen** HTML-Dokumentes, wenn das aktuelle Dokument **bereits komplett geladen**, also das Ereignis onload **bereits** ausgelöst wurde. Letzteres ist immer dann der Fall, wenn der BODY-Teil des Dokumentes komplett geparkt wurde. Grund: Ein komplett geparktes Dokument kann per write() bzw. writeln() nicht um HTML-Elemente verändert werden, da diese Methoden keine des HTML-DOM sind. Mit anderen Worten: Nur die Verwendung von Methoden des HTML-DOM lassen eine **nachträgliche** HTML-Elemente-Veränderung des Dokumentes zu.

nach der Anzeige einen Zeilenvorschub anzeigen

Plain-Text

HTML-Text

Script

Beispiel für Ersatz von document.write() durch die Eigenschaft .innerHTML (nur IE):

```
<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild">';

function Laden()
{
    ID_Div.innerHTML=Kette;           // wird sofort geparkt, also IMG-Tag ausgeführt
                                     // ( anstelle von eval()
                                     // bzw. document.write() )

    ID_IMG.SRC="";
    ID_IMG.style.display="block";

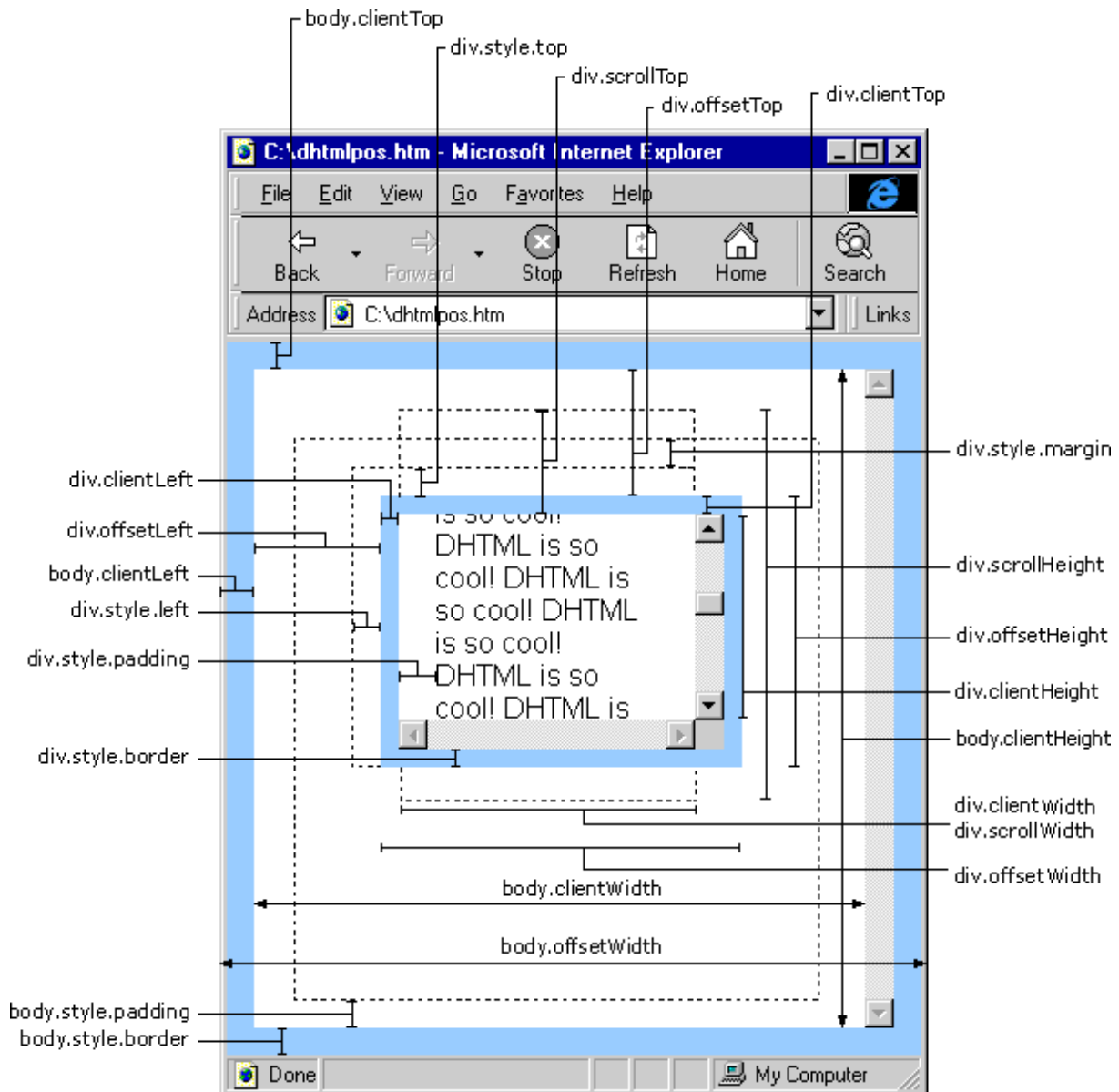
    ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
}

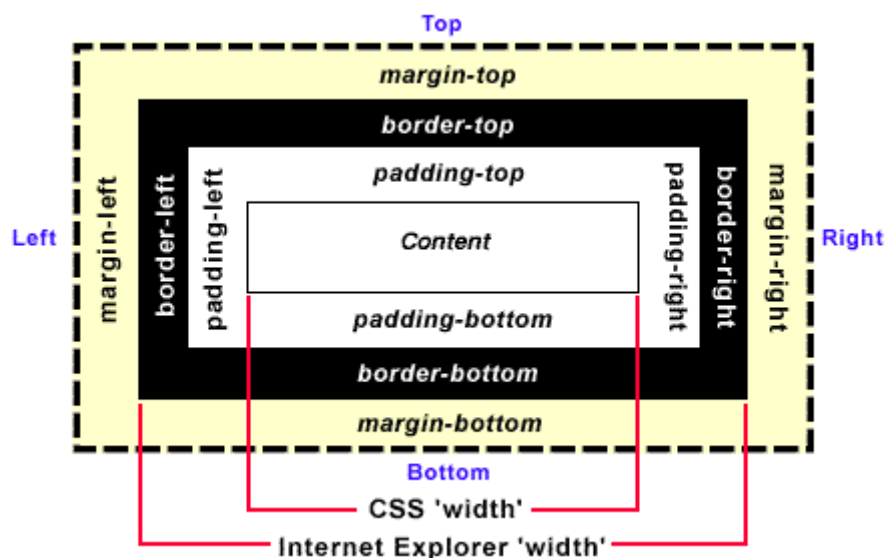
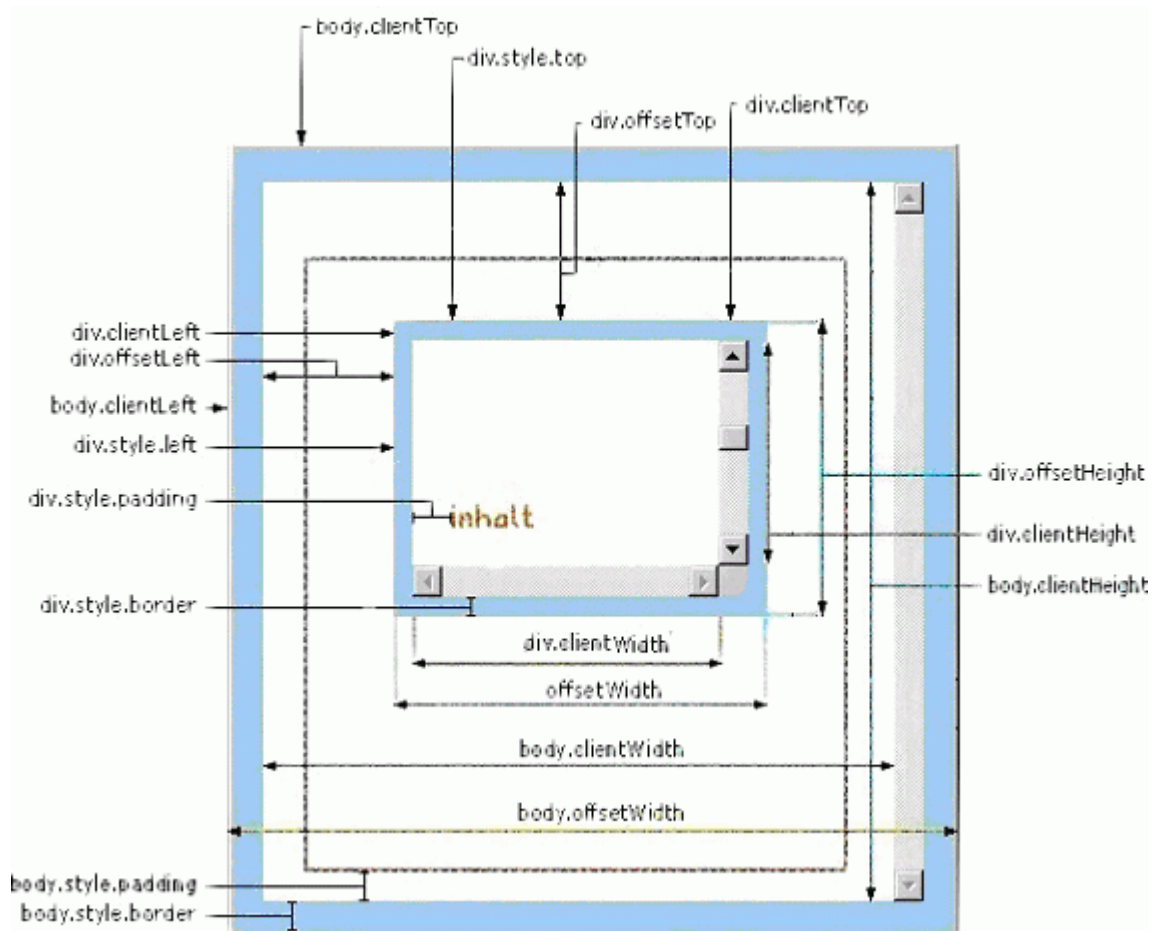
function IMGAltTextAufFehlerMeldung ()
{
    ID_IMG.alt="Das Bild konnte nicht geladen werden.";
    return true;
}
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>
```



7. Anhang: Styles des Internet Explorer

Der in nachfolgenden Beschreibungen wegen Vereinfachung verwendete Zeiger `object.style.eigenschaft` setzt voraus, dass für `object` eine konkrete Referenz kodiert werden muss.





7.1. Objekte mit STYLE-Attribut

A, ACRONYM, ADDRESS, APPLET, AREA, B, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP



7.2. *Style-Eigenschaften - Übersicht*

:active Pseudoklasse des a Objektes für aktiven Link (User hat Link aktiviert)
ist nicht per Script ansprechbar

Beispiel:

```
<HEAD>
<STYLE>
A:active { font-weight:bold; color:purple }
</STYLE>
</HEAD>
```

:first Pseudoklasse für @page Regel, siehe auch Objekt page

:first-letter Pseudoklasse für Style des ersten Buchstaben eines HTML-Elementes (Tags)
nicht in Script ansprechbar
nur für Blockelemente wie DIV und SPAN
Elemente mit Eigenschaft .style.display mit Wert "block"
ab IE 5.5

Beispiel:

```
<STYLE>
P.ErsterBuchstabe:first-letter { font-size: 200%; float: left }
</STYLE>
<P CLASS="ErsterBuchstabe">
Testtext
</P>
```

:first-line Pseudoklasse für Style der ersten Zeile eines HTML-Elementes (Tags)
nicht in Script ansprechbar
nur für Blockelemente wie DIV und SPAN
Elemente mit Eigenschaft .style.display mit Wert "block"
ab IE 5.5

Beispiel:

```
<STYLE>
P.ErsteZeile:first-letter { text-transform: uppercase }
</STYLE>
<P CLASS="ErsteZeile">
Testtext Zeile1
<BR>
Testtext Zeile2
</P>
```

:hover Pseudoklasse des a Objektes für Link, der nicht aktiviert wurde
solange Mauszeiger auf dem Link steht, ist die Klasse aktiv
ist nicht per Script ansprechbar

Beispiel:

```
<HEAD>
<STYLE>
A:hover { color:red; text-transform:uppercase; letter-spacing:1cm }
</STYLE>
</HEAD>
```

:left Pseudoklasse für @page Regel, siehe auch Objekt page

:link Pseudoklasse des a Objektes für Link, der nicht kürzlich aktiviert wurde.
ist nicht per Script ansprechbar
ab IE 4.x

Beispiel:

```
<HEAD>
<STYLE>
A:link { color:#FF0000 }
</STYLE>
</HEAD>
```

:right Pseudoklasse für @page Regel, siehe auch Objekt page

:visited Pseudoklasse des a Objektes für Link, der bereits aktiviert wurde und User ist wieder auf das Dokument
zurückgekehrt, das den Link enthält.
ist nicht per Script ansprechbar

Beispiel:

```
<HEAD>
<STYLE>
A:visited { color: blue }
</STYLE>
</HEAD>
```



.style.accelerator ALT+Taste als Kurztastenkombination-Zugriff auf Objekt ein/aus

Beispiel Label mit Tastenkombination

```
<LABEL FOR="ID_InputText">
  <U STYLE="ACCELERATOR:true">
    N
  </U>
  ame
</LABEL>
<INPUT TYPE="text"
  ID="ID_InputText"
  SIZE="25"
  ACCESSKEY="N"
  VALUE="Name"
>
```

.style.background Hintergrund eines Objektes
bei P, DIV, SPAN: Eigenschaft nur für eingeschlossenen Text

Beispiel 1

```
<HEAD>
<STYLE>
  .style1 { background:beige url(test.jpg) no-repeat top center }
  .style2 { background:ivory url(test.jpeg) no-repeat bottom right }
</STYLE>
</HEAD>
<BODY>
<SPAN onmouseover="this.className='style1'"
  onmouseout="this.className='style2'"
>
  Testtext
</SPAN>
```

Beispiel 2

```
<SPAN onclick="this.style.background='beige url(test.jpeg) no-repeat top center'">
  Testtext
</SPAN>
```

Beispiel 3 für Hintergrundbild als Wasserzeichen:

```
document.body.style.background="url('test.jpg') white center no-repeat fixed";
```

.style.backgroundAttachment Hintergrundbild
nur ab IE 4.x

Beispiel 1:

```
<HEAD>
<STYLE >
  BODY { background-attachment:fixed }
</STYLE>
</HEAD>
<BODY background="test.jpg">
.....
```

Beispiel 2:

```
<BODY ID="ID_Body"
  STYLE="background='marble05.jpg'"
  onload="ID_Body.style.backgroundAttachment = 'fixed'"
>
```

.style.backgroundColor Hintergrundfarbe
nur ab IE 4.x
Achtung: Für das body Objekt gilt:
BGColor Attribut und Eigenschaft .bgColor für die Hintergrundfarbe im BODY
sind deprecated und durch
<BODY STYLE="background-color:.....">
zu ersetzen.

Beispiel 1:

```
<SPAN STYLE="font-size:14; background-color:beige">
  Testtext
</SPAN>
```

Beispiel 2:

```
<SPAN onmouseover="this.style.backgroundColor='beige'">
```



Testtext

.style.backgroundImage Hintergrundbild, das sich über die Hintergrundfarbe legt nur ab IE 4.x

Beispiel 1:

```
<HEAD>
<STYLE>
    .setUrl { background-image: url(test.jpg) }
    .loseUrl { background-image: url(none) }
</STYLE>
</HEAD>
<BODY>
    <SPAN    STYLE="font-size:14"
            onmouseover="this.className='setUrl'"
            onmouseout="this.className='loseUrl'"
    >
        Testtext
    </SPAN>
```

Beispiel 2:

```
<SPAN onmouseover="this.style.backgroundImage='url(test.jpg)'">
    Testtext
</SPAN>
```

.style.backgroundPosition Hintergrund-Position für Hintergrund z.B. Image (linke obere Ecke)

Beispiel 1:

```
<HEAD>
<STYLE>
    .style1 { background-position:top center }
    .style2 { background-position:bottom right }
</STYLE>
</HEAD>
<BODY onload=" ID_Span.className='style1'">
    <SPAN    ID="ID_Span"
            STYLE="font-size:14; width:250;"
            onmouseover="this.className='style2'"
            onmouseout="this.className='style1'"
    >
        Testtext
    </SPAN>
```

Beispiel 2:

```
<SPAN onmouseover="this.style.backgroundPosition='bottom right'">
    Testtext
</SPAN>
```

.style.backgroundPositionX Hintergrund-Position X für Hintergrund z.B. Image (linke obere Ecke)

.style.backgroundPositionY Hintergrund-Position Y für Hintergrund z.B. Image (linke obere Ecke)

.style.backgroundRepeat Kachelung des Hintergrundbildes

Beispiel 1:

```
<HEAD>
<STYLE>
    .style1 { background-image:url(test.jpg); background-repeat:repeat }
    .style2 { background-image:url(test.jpeg); background-repeat:no-repeat }
</STYLE>
</HEAD>
<BODY>
    <SPAN    onmouseover="this.className='style1'"
            onmouseout="this.className='style2'"
            onclick="this.className=''"
    >
        Testtext
    </SPAN>
```

Beispiel 2:

```
<SPAN onmouseover="this.style.backgroundImage='url(test.jpeg)'; this.style.backgroundRepeat='repeat'">
    Testtext
</SPAN>
```

.style.behavior Ort der Behaviors-Datei *.htc (Verhaltensweise eines Elementes) bzw. Standard-Behavior des IE



Beispiel 1:

```
<ELEMENT STYLE="behavior:url(a1.htc) url(a2.htc) ..." >
```

Element steht hier als Platzhalter für ein Tag

Beispiel 2:

```
<UL>
  <LI STYLE="behavior:url(ul.htc) url(hilite.htc)">
    HTML
  </LI>
</UL>
```

Beispiel 3:

```
<HEAD>
<STYLE>
.test_style_klasse { behavior:url(ul.htc) url(hilite.htc)}
</STYLE>
</HEAD>
<BODY>
  <UL>
    <LI CLASS="test_style_klasse">
      HTML
    </LI>
  </UL>
</BODY>
```

Beispiel 4:

```
<SCRIPT>
function window.onload()
{
  ID_Li.style.behavior = "url(ul.htc) url(hilite.htc)";
}
</SCRIPT>
<UL>
  <LI ID="ID_Li" >
    HTML
  </LI>
</UL>
```

Beispiel 5:

```
<HEAD>
<STYLE>
.test_style_klasse { behavior:url(#ID_Object) }
</STYLE>
</HEAD>
<BODY>
  <OBJECT ID="ID_Object" ... ></OBJECT>
  <UL>
    <LI CLASS="test_style_klasse">
      HTML
    </LI>
  </UL>
</BODY>
```

.style.border

Rahmen alle 4 Seiten: Art und Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des
Objektes besitzen

Beispiel 1:

```
<HEAD>
<STYLE>
.applyBorder { border:0.2cm groove orange }
.removeBorder { border:none }
</STYLE>
</HEAD>
<BODY>
  <TABLE BORDER>
  <TR>
    <TD
      onmouseover="this.className='applyBorder'"
      onmouseout="this.className='removeBorder'"
    >
      <IMG SRC="test.jpg">
    </TD>
  </TR>
```



</TABLE>

Beispiel 2:

<TD onmouseover="this.style.border='0.2cm groove pink'">

.style.borderBottom

Rahmen unten: Art und Dicke
 vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```
<HEAD>
<STYLE>
    TD { border-bottom:0.5cm solid yellow }
    .change { border-bottom:0.5cm groove pink }
</STYLE>
</HEAD>
<BODY>
    <TABLE>
    <TR>
        <TD
            onmouseover="this.className='change'"
            onmouseout="this.className=''"
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>
</BODY>
```

Beispiel 2:

<TD onmouseover="this.style.borderBottom='0.3cm groove yellow'">

.style.borderBottomColor

Rahmen unten: Farbe
 vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```
<HEAD>
<STYLE>
    TD { border-bottom-color: red; border-width: 0.5cm; border-style: groove }
    .blue { border-bottom-color: blue }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
    <TR>
        <TD
            onmouseover="this.className='blue'"
            onmouseout="this.className=''"
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>
</BODY>
```

Beispiel 2:

<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderBottomColor='blue'">

.style.borderBottomStyle

Rahmen unten: Art
 vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```
<HEAD>
<STYLE>
    TD { border-bottom-style:solid; border-width=0.3cm }
    .change { border-bottom-style:groove }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
    <TR>
        <TD
            onmouseover="this.className='change'"
            onmouseout="this.className=''"
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
```



</TABLE>

Beispiel 2:

<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderBottomStyle='groove'">

.style.borderBottomWidth

Rahmen unten:

Dicke

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

<HEAD>

<STYLE>

TD { border-bottom-width:3mm }

.changeborder1 { border-bottom-width:1cm }

</STYLE>

</HEAD>

<BODY>

<TABLE BORDER>

<TR>

<TD

onclick="this.className='changeborder1'"

ondblclick="this.className=""

>

</TD>

</TR>

</TABLE>

</BODY>

Beispiel 2:

<TD onclick="this.style.borderBottomWidth='1cm'">

.style.borderCollapse

Rahmen alle 4 Seiten:

Verschmelzung zum Single Border

Beispiel:

<TABLE ID="ID_Tabelle" STYLE="border-collapse:collapse">

.....

</TABLE>

<INPUT TYPE=button VALUE="separate" onclick="ID_Tabelle.style.borderCollapse='separate'">

<INPUT TYPE=button VALUE="collapse" onclick="ID_Tabelle.style.borderCollapse='collapse'">

.style.borderColor

Rahmen alle 4 Seiten:

Farbe

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

<HEAD>

<STYLE>

TD { border-color: red; border-width: 0.5cm }

.blue { border-color: blue }

</STYLE>

</HEAD>

<BODY>

<TABLE BORDER>

<TR>

<TD

onmouseover="this.className='blue'"

onmouseout="this.className=""

>

</TD>

</TR>

</TABLE>

</BODY>

Beispiel 2:

<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderColor='blue'; this.style.borderStyle='solid'">

.style.borderLeft

Rahmen links:

Art und Dicke

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

<HEAD>

<STYLE>

TD { border-left:0.5cm solid yellow }

.change { border-left:0.5cm groove pink }

</STYLE>

</HEAD>

<BODY>




```

<TABLE>
<TR>
  <TD    onmouseover="this.className='change'"
        onmouseout="this.className=''"
  >
    <IMG SRC="test.jpg">
  </TD>
</TR>
</TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderLeft='0.3cm groove yellow'">
```

`.style.borderLeftColor` Rahmen links: Farbe
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
  TD { border-left-color: red; border-width: 0.5cm; border-style: groove }
  .blue { border-left-color: blue }
</STYLE>
</HEAD>
<BODY>
  <TABLE BORDER>
    <TR>
      <TD    onmouseover="this.className='blue'"
            onmouseout="this.className=''"
      >
        <IMG SRC="test.jpg">
      </TD>
    </TR>
  </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderLeftColor='blue'">
```

`.style.borderLeftStyle` Rahmen links: Art
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
  TD { border-left-style:solid; border-width:0.3cm }
  .change { border-left-style:groove }
</STYLE>
</HEAD>
<BODY>
  <TABLE BORDER>
    <TR>
      <TD    onmouseover="this.className='change'"
            onmouseout="this.className=''"
      >
        <IMG SRC="test.jpg">
      </TD>
    </TR>
  </TABLE>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderLeftStyle='groove'">
```

`.style.borderLeftWidth` Rahmen links: Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
  TD { border-left-width:3mm }
  .changeborder1 { border-left-width:1cm }
</STYLE>
</HEAD>
<BODY>
  <TABLE BORDER>

```



```

        <TR>
            <TD
                onclick="this.className='changeborder1'"
                ondblclick="this.className="
            >
                <IMG SRC="test.jpg">
            </TD>
        </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onclick="this.style.borderLeftWidth='1cm'">
```

.style.borderRight

Rahmen rechts: Art und Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-right:0.5cm solid yellow }
    .change { border-right:0.5cm groove pink }
</STYLE>
</HEAD>
<BODY>
    <TABLE>
    <TR>
        <TD
            onmouseover="this.className='change'"
            onmouseout="this.className="
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderRight='0.3cm groove yellow'">
```

.style.borderRightColor

Rahmen rechts: Farbe
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-right-color: red; border-width: 0.5cm; border-style: groove }
    .blue { border-right-color: blue }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
    <TR>
        <TD
            onmouseover="this.className='blue'"
            onmouseout="this.className="
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderRightColor='blue'">
```

.style.borderRightStyle

Rahmen rechts: Art
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-right-style:solid; border-width=0.3cm }
    .change { border-right-style:groove }
</STYLE>
</HEAD>
<BODY>

```



```

<TABLE BORDER>
<TR>
    <TD    onmouseover="this.className='change'"
           onmouseout="this.className=''"
    >
        <IMG SRC="test.jpg">
    </TD>
</TR>
</TABLE>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderRightStyle='groove'">
```

.style.borderRightWidth Rahmen rechts: Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-right-width:3mm }
    .changeborder1 { border-right-width:1cm }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
    <TR>
        <TD    onclick="this.className='changeborder1'"
               ondblclick="this.className=''"
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onclick="this.style.borderRightWidth='1cm'">
```

.style.borderStyle Rahmen alle 4 Seiten: Art
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-style:solid; border-width:0.3cm }
    .change { border-style:groove }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
    <TR>
        <TD    onmouseover="this.className='change'"
               onmouseout="this.className=''"
        >
            <IMG SRC="test.jpg">
        </TD>
    </TR>
    </TABLE>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderStyle='groove'">
```

.style.borderTop Rahmen oben: Art und Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-top:0.5cm solid yellow }
    .change { border-top:0.5cm groove pink }
</STYLE>
</HEAD>
<BODY>
    <TABLE>
    <TR>
        <TD    onmouseover="this.className='change'"

```



```

        onmouseout="this.className=""
    >
    <IMG SRC="test.jpg">
</TD>
</TR>
</TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderTop=0.3cm groove yellow">
```

.style.borderTopColor

Rahmen oben:

Farbe

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-top-color: red; border-width: 0.5cm; border-style: groove }
    .blue { border-top-color: blue }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
        <TR>
            <TD
                onmouseover="this.className='blue'"
                onmouseout="this.className=""
            >
                <IMG SRC="test.jpg">
            </TD>
        </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderTopColor='blue'">
```

.style.borderTopStyle

Rahmen oben:

Art

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-top-style:solid; border-width=0.3cm }
    .change { border-top-style:groove }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
        <TR>
            <TD
                onmouseover="this.className='change'"
                onmouseout="this.className=""
            >
                <IMG SRC="test.jpg">
            </TD>
        </TR>
    </TABLE>

```

Beispiel 2:

```
<TD onmouseover="this.style.borderWidth='0.5cm'; this.style.borderTopStyle='groove'">
```

.style.borderTopWidth

Rahmen oben:

Dicke

vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-top-width:3mm }
    .changeborder1 { border-top-width:1cm }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
        <TR>
            <TD
                onclick="this.className='changeborder1'"
                ondblclick="this.className=""
            >

```



```

        >
        <IMG SRC="test.jpg">
    </TD>
</TR>
</TABLE>
</BODY>

```

Beispiel 2:

```
<TD onclick="this.style.borderTopWidth='1cm'">
```

.style.borderWidth Rahmen alle 4 Seiten: Dicke
vor IE 5.5: Objekt muss absolut positionierte sein, oder Angabe zu Höhe bzw. Breite des Objektes besitzen

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { border-width:3mm }
    .changeborder1 { border-width:1cm }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
        <TR>
            <TD
                onclick="this.className='changeborder1'"
                ondblclick="this.className=''"
            >
                <IMG SRC="test.jpg">
            </TD>
        </TR>
    </TABLE>
</BODY>

```

Beispiel 2:

```
<TD onclick="this.style.borderWidth='1cm'">
```

.style.bottom Abstand unterer Objektrand zur Umgebung Oberkante
Objekt muss position-Eigenschaft kodiert haben
identisch mit Attribut **.style.pixelBottom** und **.style.posBottom**
aber **.style.bottom**: String
 .style.pixelBottom: Integer, nur Pixel
 .style.posBottom: Float pointing und Integer

Beispiel:

```

<DIV STYLE = "position:absolute; bottom:50px">
    ...
</DIV>

```

.style.clear Textumfluss links, rechts ("left", "right", "both", "none")
siehe auch Eigenschaften **.style.whiteSpace**
 .style.float
 .style.styleFloat

Beispiel 1:

```

<HEAD>
<STYLE>
    I { clear:left }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<HEAD>
<SCRIPT>
    function Wechsel1()
    { ID_Span.style.clear="left";}

    function Wechsel2()
    { ID_Span.style.clear="none";}
</SCRIPT>
</HEAD>
<BODY>
    <IMG SRC ="test.jpg" STYLE="float:left">
    <SPAN ID="ID_Span">
        TestText
    </SPAN>
    <INPUT TYPE=button VALUE="clear = left" onclick="Wechsel1()">

```



```

        <INPUT TYPE=button VALUE="clear = none" onclick="Wechsel2()">
    </BODY>

```

.style.clip

Position sichtbarer Bereich über dem Objekt
 Position immer bezüglich Umgebung
 Objekt muss absolut positioniert sein !
 ausserhalb des Clipfensters wird alles transparent
 siehe auch Eigenschaften

- .style.overflow
- .style.overflowX
- .style.overflowY
- .style.display
- .style.visibility

Beispiel 1:

```

<DIV STYLE="position:absolute;top:0;left:200px; clip:rect(15px auto auto auto)">
    <IMG SRC="test.jpg">
</DIV>

```

Beispiel 2:

```

<IMG ID="ID_IMG" SRC="test.jpg" STYLE="position:absolute;top:0cm;left:0cm;">
<BUTTON onclick="ID_IMG.style.clip='rect(0.2cm 0.6cm 1cm 0.1cm)'">
    Sichtbarkeitsbereich über Image
</BUTTON>

```

.style.color Vordergrundfarbe (Textfarbe)

Beispiel 1:

```

<HEAD>
<STYLE>
    .color1 { color:red }
    .color2 { color:gray }
</STYLE>
</HEAD>
<BODY>
    <SPAN     STYLE="font-size:14"
        onmouseover="this.className='color1'"
        onmouseout="this.className='color2'"
    >
        Testtext
    </SPAN>
</BODY>

```

Beispiel 2:

```

<SPAN STYLE="font-size:14" onmouseover="this.style.color='red'">
    Testtext
</SPAN>

```

Beispiel 3 Überschrift mit permanentem Farbwechsel:

```

<HTML>
<HEAD>
<STYLE TYPE="text/css">
<!--
    # farbwechsel { color: #FF0000; }
-->
</STYLE>

<SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
<!--
    var farbwechsel_anzahl      = 32;           // Anzahl der Schritte von Start bis Ende
    var farbwert_start          = [255, 0, 0]   // Rot, Grün, Blau
    var farbwert_ende           = [0, 255, 255] // Rot, Grün, Blau
    var farbwechsel_tempo       = 25           // in ms zwischen zwei Schritten

    var farbwechsel_zahler      = 0
    var farbwert_wechselrichtung = 0

    var hexaziffern_feld =
    [
        "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
        "A", "B", "C", "D", "E", "F"
    ];

    function dezimalwert_zu_hexaziffern(dezimal_wert)
    {
        var high = Math.floor(dezimal_wert / 16);

```



```

        var low = Math.floor(dezimal_wert - (high * 16));

        return(hexaziffern_feld[high] + "" + hexaziffern_feld[low]);
    }

    function farbewechseln(ueberschrift_id)
    {
        var i;
        var farbe_dezimal = new Array(3);           // Rot, Grün, Blau

        for(i = 0; i < 3; i++)
        {
            farbe_dezimal[i] =
                farbwert_start[i]
                - (
                    (farbwert_start[i] - farbwert_ende[i])
                    * farbwechsel_zahler
                    / f arbwechsel_anzahl
                )
        }

        if(farbwechsel_zahler >= farbwechsel_anzahl)
        {farbwert_wechselrichtung = -1}
        else
        {
            if(farbwechsel_zahler <= 1)
            {farbwert_wechselrichtung = 1}
        }

        farbwechsel_zahler += farbwert_wechselrichtung

        var farbe_hexa =
            "#"
            + dezimalwert_zu_hexaziffern(farbe_dezimal[0])
            + dezimalwert_zu_hexaziffern(farbe_dezimal[1])
            + dezimalwert_zu_hexaziffern(farbe_dezimal[2]);

        if(document.ids)
        {document.ids[ueberschrift_id].color = farbe_hexa}           // NS
        else
        {document.all[ueberschrift_id].style.color = farbe_hexa}     // IE

        id = window.setTimeout(
            "farbewechseln('" + ueberschrift_id + "')", farbwechsel_tempo);
        );
    }

    function start(
        ueberschrift_id,
        startfarbe_rot, startfarbe_gruen, startfarbe_blaue,
        endfarbe_rot, endfarbe_gruen, endfarbe_blaue
    )
    {
        farbwert_start[0] = startfarbe_rot;
        farbwert_start[1] = startfarbe_gruen;
        farbwert_start[2] = startfarbe_blaue;
        farbwert_ende[0] = endfarbe_rot;
        farbwert_ende[1] = endfarbe_gruen;
        farbwert_ende[2] = endfarbe_blaue;
        farbwechsel_zahler = 0;
        farbewechseln(ueberschrift_id);
    }
}
//-->
</SCRIPT>
</HEAD>

<BODY onLoad="start('farbwechsel', 0, 255, 0, 255, 0, 255)">
    <H1 ID="farbwechsel">&Uuml;berschrift</H1>
</BODY>
</HTML>

```

..style.cssText Wert des STYLE-Attributes im Tag des HTML-Elementes

Beispiel:

```

<P ID="ID_P" STYLE="color:'green'; font-weight:bold">
    Testtext

```

```

</P>

```

```

<BUTTON onclick="alert(ID_P.style.cssText)">
    Anzeigen

```




```

function Aendern()
{
    ID_Zelle1.style.display="none";
    ID_Zelle2.style.display="block";
    ID_Zelle3.style.display="none";
}
</SCRIPT>
<TABLE>
  <TR>
    <TD ID="ID_Zelle1">
      Zelle1
    </TD>
  </TR>
  <TR>
    <TD ID="ID_Zelle2">
      Zelle2
    </TD>
  </TR>
  <TR>
    <TD ID="ID_Zelle3">
      Zelle3
    </TD>
  </TR>
</TABLE>
<INPUT TYPE=button VALUE="Aendern" onclick="Aendern()">

```

.style.filter Filter des Internet Explorer implementieren (siehe Objekt filter)

Beispiel Filter lesen

```

<SCRIPT>
  alert(ID_Div.style.filter);
  // zeigt den String
  //      "filter:progid:DXImageTransform.Microsoft.Wave(strength=100)
  //      progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)"
  // an
</SCRIPT>
<DIV ID="ID_Div" STYLE="height:50;width:50;
  filter:progid:DXImageTransform.Microsoft.Wave(strength=100)
  progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)"
>
  Test-Text
</DIV>

```

Beispiel Filter schreiben

```

<DIV ID="ID_Div" STYLE="height:50;width:50;filter:
  progid:DXImageTransform.Microsoft.Wave(strength=100)
  progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)"
>
  Test-Text
</DIV>
<SCRIPT>
  // hinzufügen
  ID_Div.style.filter += "progid:DXImageTransform.Microsoft.Iris(irisstyle='STAR',duration=4)"
</SCRIPT>

```

float Textumfluss links, rechts
 nur im STYLE-Attribut des HTML-Elementes kodierbar
 per Script nicht ansprechbar
 unter IE 5.x : DIV und SPAN müssen Eigenschaft .style.width besitzen
 siehe auch Eigenschaften .style.whiteSpace
 .style.clear
 .style.styleFloat

Beispiel:

```
<IMG SRC="test.jpg" STYLE="float:left">
```

.style.font Textfont komplett mit allen Teileigenschaften (ebenfalls Style-Eigenschaften) oder Standard-Textfont

Beispiel 1:

```

<SPAN STYLE="font:italic normal bolder 12pt Arial">
  Testtext
</SPAN>

```



Beispiel 2

```
<DIV onmouseover="this.style.font = 'italic small-caps bold 12pt serif'">
  Testtext
</DIV>
```

.style.fontFamily Textfont Familie

Beispiel 1:

```
<HEAD>
<STYLE>
  P { font-family:"Arial" }
  .other { font-family:"Courier" }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<DIV onmousedown="this.style.fontFamily='Courier'">
```

.style.fontSize Textfont Höhe

Beispiel 1:

```
<HEAD>
<STYLE>
  BODY { font-size: 10pt }
  .P1 { font-size: 14pt }
  .P2 { font-size: 75% }
  .P3 { font-size: xx-large }
  .P4 { font-size: larger }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<DIV STYLE="font-size:12pt" onmouseover="this.style.fontSize='14pt'">
  Testtext
</DIV>
```

.style.fontStyle Textfont Stil

Beispiel 1:

```
<HEAD>
<STYLE>
  H3 { font-style:italic }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<DIV onmousedown="this.style.fontStyle='italic'">
```

.style.fontVariant Textfont Stil-Variante

Beispiel 1:

```
<P STYLE="font-variant:small-caps">
```

Beispiel 2:

```
<DIV onmousedown="this.style.fontVariant='small-caps'">
```

.style.fontWeight Textfont Fetttheit

Beispiel 1:

```
<HEAD>
<STYLE>
  LI { font-weight:bolder }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<P STYLE="font-size:14" onmouseover="this.style.fontWeight='bolder'">
```

.style.height Höhe Objekt
identisch mit .style.pixelHeight und .style.scrollHeight, aber

.style.height:	String
.style.pixelHeight:	Integer, nur Pixel
.style.scrollHeight:	Float pointing und Integer

Hinweis: Wenn !DOCTYPE am Dokumentanfang kodiert wurde, so **kann**
.style.height auf "auto "



gesetzt sein
 Wenn **nicht** !DOCTYPE am Dokumentanfang kodiert wurde, so **darf**
 .style.height **nicht** auf "auto "
 gesetzt sein
!DOCTYPE am Dokumenten-Anfang ist erst ab IE 6.x kodierbar !

Beispiel 1:

```
<IMG SRC="test.jpg" STYLE="height:4cm">
```

Beispiel 2:

```
<BUTTON onclick="height1.style.height='1cm'">Test</BUTTON>
```

.style.imeMode Status des Input Method Editor (IME) für Eingabe von Chinese, Japanese oder Korean Zeichen

Beispiel:

```
<INPUT TYPE = text STYLE = "ime-mode:active" >
```

.style.layoutFlow deprecated
 zu ersetzen durch Eigenschaften .style.writingMode
 Ost-Asiatische Darstellung (Flussrichtung)
 siehe auch Eigenschaft .style.verticalAlign für Ausrichtung vertikal für Objekte mit VALIGN-Attribut

.style.layoutGrid Textzeichen-Layout-Gitter z.B. für asiatische Sprachen

Beispiel:

```
<HEAD>
<STYLE>
  DIV.layout { layout-grid: both fixed 12px 12px }
</STYLE>
</HEAD>
<BODY>
  <DIV CLASS = "layout">
    Testtext
  </DIV>
</BODY>
```

.style.layoutGridChar Textzeichen-Layout-Gitter Zeichengröße
 nur für Blockelemente wie SPAN und DIV etc.
 verlangt .style.layoutGridMode auf Wert "line" oder "both"

Beispiel:

```
<HEAD>
<STYLE>
  DIV.layout { layout-grid-char: auto }
</STYLE>
</HEAD>
<BODY>
  <DIV CLASS = "layout">
    Testtext
  </DIV>
</BODY>
```

.style.layoutGridLine Textzeichen-Layout-Gitter Linie
 nur für Blockelemente wie SPAN und DIV etc.
 verlangt .style.layoutGridMode auf Wert "line" oder "both"

Beispiel:

```
<HEAD>
<STYLE>
  DIV.layout { layout-grid-line: auto }
</STYLE>
</HEAD>
<BODY>
  <DIV CLASS = "layout">
    Testtext
  </DIV>
</BODY>
```

.style.layoutGridMode Textzeichen-Layout-Gitter 2D

Beispiel:

```
<HEAD>
<STYLE>
  DIV.layout { layout-grid-mode: line }
</STYLE>
</HEAD>
<BODY>
```



```

<DIV CLASS = "layout">
    Testtext
</DIV>
<BODY>

```

`.style.layoutGridType` Textzeichen-Layout-Gitter Typ
nur für Block-Elemente wie SPAN, DIV

Beispiel:

```

<HEAD>
<STYLE>
    DIV.layout { layout-grid-type: strict }
</STYLE>
</HEAD>
<BODY>
    <DIV CLASS = "layout">
        Testtext
    </DIV>
</BODY>

```

`.style.left` Abstand linker Objektrand zur Umgebung rechte Kante
Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes,
wenn im Elternelement `STYLE="position:....."` **kodiert wurde** (Wert von position ist egal)
Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):
Abstand der linken Kante des IMG zur linken Kante des DIV
Objekt muss Eigenschaft `.style.position` kodiert haben
identisch zu `.style.pixelLeft` und `.style.posLeft`
aber `.style.left`: String
 `.style.pixelLeft`: Integer, nur Pixel
 `.style.posLeft`: Float pointing und Integer

Beispiel 1:

```

<DIV STYLE="position:absolute;left:100px">
    <IMG SRC="test.jpg">
</DIV>

```

Beispiel 2:

```

<BUTTON onclick="cone.style.left='100px'; sphere.style.left='200px'">
    ...
</BUTTON>

```

`.style.letterSpacing` Textzeichen Abstand

Beispiel 1:

```

<HEAD>
<STYLE>
    BLOCKQUOTE { letter-spacing:-0.2mm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<DIV STYLE="font-size:14" onmouseover="this.style.letterSpacing='1mm'">
    ....
</DIV>

```

`.style.lineBreak` Zeilenumbruch normal

`.style.lineHeight` Abstand zweier Objekte oder zweier Textzeilen
ab IE 4.x

Beispiel 1:

```

<HEAD>
<STYLE>
    P { line-height:8mm}
    BLOCKQUOTE { line-height:4mm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<DIV STYLE="font-size:14" onmouseover="this.style.lineHeight='6mm'">
    .....
</DIV>

```

`.style.listStyle` Listendarstellung Aufzählungsliste-Elemente
benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten

Beispiel 1:

```

<HEAD>

```



```

<STYLE>
    UL { list-style: outside url(test.gif) }
    UL.compact { list-style-image:none; list-style: inside circle }
</STYLE>
</HEAD>
<BODY>
    <UL CLASS=compact>
        .....
    </UL>
</BODY>

```

Beispiel 2:

```
<UL onmouseover="this.style.listStyle='url(test.gif) circle'">
```

.style.listStyleImage Listendarstellung Marker als Image
 siehe auch **.style.listStyleType**
 benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten

Beispiel 1:

```

<HEAD>
<STYLE>
    UL { list-style-image:url(test.gif) }
</STYLE>
</HEAD>

```

Beispiel 2:

```
<UL onmouseover="this.style.listStyleImage='url(test.gif)'">
```

.style.listStylePosition Listendarstellung Marker Position
 benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten

Beispiel 1:

```

<HEAD>
<STYLE>
    UL { list-style-position:inside }
    UL.compact { list-style-position:outside }
</STYLE>
</HEAD>
<BODY>
    <UL CLASS=compact>
        .....
    </UL>

```

Beispiel 2:

```

<SPAN STYLE="width:3cm"
onmouseover="this.style.listStylePosition='inside'"
onmouseout="this.style.listStylePosition='outside'"
>

```

.style.listStyleType Listendarstellung Markertyp (nicht Image)
 siehe auch **.style.listStyleTypeImage**
 benötigt linken Abstand zum linken Objekt von mindestens 30 Bildpunkten

Beispiel 1:

```

<HEAD>
<STYLE>
    UL { list-style-type:circle }
</STYLE>
</HEAD>

```

Beispiel 2:

```
<UL onmouseover="this.style.listStyleType='circle'">
```

.style.margin Aussenrand Abstand des Aussenrandes links, rechts, oben, unten des Objektes
 von anderen Objekten
 im Abstandsbereich wird immer transparent angezeigt
 IE 4.x Angaben nicht möglich für TD und TR
 unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und
 Breite besitzen
 Hinweise: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objekthinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    IMG { margin:1cm }
</STYLE>

```



Beispiel 2:

```
<IMG SRC="test.jpg" onmouseover="this.style.margin='5mm'">
```

.style.marginBottom Aussenrand Abstand unten zum angrenzenden Objekt
im Abstandsbereich wird immer transparent angezeigt
IE 4.x Angaben nicht möglich für TD und TR
unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen
Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```
<HEAD>
<STYLE>
    IMG { margin-bottom:2cm }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<IMG SRC="test.jpg"onmouseover="this.style.marginBottom='1cm'">
```

.style.marginLeft Aussenrand Abstand links zum angrenzenden Objekt
im Abstandsbereich wird immer transparent angezeigt
IE 4.x Angaben nicht möglich für TD und TR
unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen
Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```
<HEAD>
<STYLE>
    IMG { margin-left:2cm }
</STYLE>
</HEAD>
```

Beispiel 2:

```
<IMG SRC="test.jpg" onclick="this.style.marginLeft='1cm'">
```

.style.marginRight Aussenrand Abstand rechts zum angrenzenden Objekt
im Abstandsbereich wird immer transparent angezeigt
IE 4.x Angaben nicht möglich für TD und TR
unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen
Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```
<HEAD>
<STYLE>
    IMG { margin-right:1cm }
    .margin1 { margin-right:2cm }
</STYLE>
</HEAD>
<BODY>
    <IMG SRC="test.jpg" onclick="this.className='margin1'"
        ondblclick="this.className=""
    >
</BODY>
```

Beispiel 2

```
<IMG SRC="test.jpeg" onclick="this.style.marginRight='1cm'">
```

.style.marginTop Aussenrand Abstand oben zum angrenzenden Objekt
im Abstandsbereich wird immer transparent angezeigt
IE 4.x Angaben nicht möglich für TD und TR
unter IE 5.5 nur für Elemente die absolut positioniert sind, oder Angaben zu Höhe und Breite besitzen
Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1

```
<HEAD>
<STYLE>
    HR { margin-top:2cm }
    .margin1 { margin-top:4cm }
</STYLE>
</HEAD>
```



```

<BODY>
  <HR onclick="this.className='margin1'" ondblclick="this.className=''">
</STYLE>

```

Beispiel 2:

```

<HR
  onclick="this.style.marginTop='2cm'"
  ondblclick="this.style.marginTop=''"
>

```

.style.minHeight minimale Höhe des Objektes
 ab IE 6.x **und** nur wenn **nicht** !DOCTYPE am Dokumentanfang kodiert wurde
 und dann auch nur für TD, TH und TR innerhalb eines fixierten Tabellenlayouts
 Tabelle mit fixiertem Layout erzeugen per Eigenschaft .style.tableLayout mit Wert "fixed"
 wird schneller dargestellt als bei Auto-Layout
 Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto"
 also Auto-Layout, also kein fixiertes Tabellenlayout

.style.onOffBehavior deprecated ab IE 5.x
 Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound

.style.overflow Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster,
 also Überlauf der Anzeige gegenüber Elternfenster.
 Anzeige des Objektes mit/ohne Scrollelemente vertikal und horizontal
 Anzeige des Objektes ein/aus
 siehe auch Eigenschaften .style.overflowX
 .style.overflowY
 .style.clip
 .style.display
 .style.visibility
 .style.textOverflow
 ab IE 5.x

Beispiel 1

```

<DIV STYLE="width: 200px; height: 200px; overflow: auto;">
  Testtext
</DIV>

```

Beispiel 2

```

<SCRIPT>
  function AnzeigeArtSetzen(ObjektZeiger, AnzeigeArtAlsKette)
  {
    ObjektZeiger.style.overflow = AnzeigeArtAlsKette;
  }
</SCRIPT>
<DIV ID="ID_Div" STYLE="font-size:18pt;background-color:yellow;height:50px;width:75px">
  Testtext
</DIV>
<SELECT onchange="AnzeigeArtSetzen(ID_Div, this.options[this.selectedIndex].text)">
  <OPTION SELECTED>visible
  <OPTION>scroll
  <OPTION>hidden
  <OPTION>auto
</SELECT>

```

.style.overflowX Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster,
 also Überlauf der Anzeige gegenüber Elternfenster.
 Anzeige des Objektes mit/ohne Scrollelemente nur horizontal
 Anzeige des Objektes ein/aus
 siehe auch Eigenschaften .style.overflow
 .style.overflowY
 .style.clip
 .style.display
 .style.visibility
 .style.textOverflow
 ab IE 5.x

.style.overflowY Anzeige des Objektes, wenn Objektdimension in Höhe und/ oder Breite größer als das Elternfenster,
 also Überlauf der Anzeige gegenüber Elternfenster.
 Anzeige des Objektes mit/ohne Scrollelemente nur vertikal
 Anzeige des Objektes ein/aus
 siehe auch Eigenschaften .style.overflow
 .style.overflowX
 .style.clip
 .style.display
 .style.visibility
 .style.textOverflow



ab IE 5.x

`.style.padding`

Abstand links, rechts, oben, unten zwischen Objekt und Margin bzw. Rahmen

ab IE 6.x auch für `img` Objekt

unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { padding:3mm 8mm }
    .padding1 { padding:1cm }
</STYLE>
</HEAD>
<BODY>
    <TABLE BORDER>
        <TD ALIGN=middle
            onmouseover="this.className='padding1'"
            onmouseout="this.className=''"
            >
                <IMG SRC="test.jpg">
        </TD>
    </TABLE>
</BODY>

```

Beispiel 2:

```

<TABLE BORDER>
    <TD ALIGN=middle
        onmouseover="this.style.padding='0.5cm 0.2cm'"
        onmouseout="this.style.padding=''"
        >
            <IMG SRC="test.jpeg">
    </TD>
</TABLE>

```

`.style.paddingBottom`

Abstand unten zwischen Objekt und Margin bzw. Rahmen

ab IE 6.x auch für `img` Objekt

unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { padding-bottom:1cm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<TABLE BORDER>
    <TD onmouseover="this.style.paddingBottom='1cm'"
        onmouseout="this.style.paddingBottom=''"
        >
            <IMG SRC="test.jpeg">
    </TD>
</TABLE>

```

`.style.paddingLeft`

Abstand links zwischen Objekt und Margin bzw. Rahmen

ab IE 6.x auch für `img` Objekt

unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben

Margin: Abstand des Aussenrandes eines Objektes zur Umgebung

Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { padding-left:1cm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<TABLE BORDER>
    <TD onmouseover="this.style.paddingLeft='1cm'"
        onmouseout="this.style.paddingLeft=''"
        >

```




```

        <IMG SRC="test.jpeg">
    </TD>
</TABLE>

```

.style.paddingRight Abstand rechts zwischen Objekt und Margin bzw. Rahmen
 ab IE 6.x auch für img Objekt
 unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben
 Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { padding-right: 1cm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<TABLE BORDER>
    <TD onmouseover="this.style.paddingRight='1cm'"
        onmouseout="this.style.paddingRight=''"
        >
        <IMG SRC="test.jpeg">
    </TD>
</TABLE>

```

.style.paddingTop Abstand oben zwischen Objekt und Margin bzw. Rahmen
 ab IE 6.x auch für img Objekt
 unter IE 5.5 nur für Elemente die absolut positioniert sind oder Angaben zu Höhe und Breite haben
 Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objektinhaltes zum Aussenrand

Beispiel 1:

```

<HEAD>
<STYLE>
    TD { padding-top: 1cm }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<TABLE BORDER>
    <TD onmouseover="this.style.paddingTop='1cm'"
        onmouseout="this.style.paddingTop=''"
        >
        <IMG SRC="test.jpeg">
    </TD>
</TABLE>

```

.style.pageBreakAfter Seitenumbruch nach dem Druck des Objektes beim Druck des Dokumentes
 nicht möglich für BR und HR-Objekt
 siehe auch Eigenschaften `.style.whiteSpace`
`.style.wordBreak`
`.style.wordWrap`
`.style.pageBreakBefore`

Beispiel 1:

```

<HEAD>
<STYLE>
    P { page-break-after: always }
</STYLE>
</HEAD>
<BODY>
<P ID="ID_P">
    .....
</P>

```

Beispiel 2:

```

<SCRIPT>
    function KeinSeitenumbruch()
    { ID_P.style.pageBreakAfter=''; }
</SCRIPT>
<BUTTON onClick="KeinSeitenumbruch()">Kein Seitenumbruch </BUTTON>

```

.style.pageBreakBefore Seitenumbruch vor dem Druck des Objektes beim Druck des Dokumentes
 nicht möglich für BR und HR-Objekt
 siehe auch Eigenschaften `.style.whiteSpace`
`.style.wordBreak`



```
.style.wordWrap
.style.pageBreakAfter
```

Beispiel 1:

```
<HEAD>
<STYLE>
    P { page-break-before: always }
</STYLE>
</HEAD>
<BODY>
<P ID="ID_P">
    .....
</P>
```

Beispiel 2:

```
<SCRIPT>
    function KeinSeitenumbruch()
    {ID_P.style.pageBreakBefore="";}
</SCRIPT>
<BUTTON onClick="KeinSeitenumbruch()">Kein Seitenumbruch </BUTTON>
```

.style.pixelBottom	<p>Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und style.posBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer</p>
.style.pixelHeight	<p>Objekthöhe identisch mit .style.height und .style.posHeight aber .style.height: String .style.pixelHeight: Integer, nur Pixel .style.posHeight: Float pointing und Integer</p>
.style.pixelLeft	<p>Abstand linker Objektrand zur Umgebung rechte Kante Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes, wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal) Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV): Abstand der linken Kante des IMG zur linken Kante des DIV identisch mit Attribut .style.left und .style.posLeft aber .style.left: String .style.pixelLeft: Integer, nur Pixel .style.posLeft: Float pointing und Integer</p>
.style.pixelRight	<p>Abstand rechter Objektrand zur Umgebung linke Kante identisch mit Attribut .style.right und .style.posRight aber .style.right: String .style.pixelRight: Integer, nur Pixel .style.posRight: Float pointing und Integer</p>
.style.pixelTop	<p>Abstand oberer Objektrand zur Umgebung Unterkante Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes, wenn im Elternelement STYLE="position:....." kodiert wurde (Wert von position ist egal) Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV): Abstand der oberen Kante des IMG zur oberen Kante des DIV identisch mit Attribut .style.top und .style.posTop aber .style.top: String .style.pixelTop: Integer, nur Pixel .style.posTop: Float pointing und Integer</p>
.style.pixelWidth	<p>Objektbreite identisch mit .style.width und .style.posWidth aber .style.width: String .style.pixelWidth: Integer, nur Pixel .style.posWidth: Float pointing und Integer</p>
.style.posBottom	<p>Abstand unterer Objektrand zur Umgebung Oberkante identisch mit Attribut .style.bottom und .style.pixelBottom aber .style.bottom: String .style.pixelBottom: Integer, nur Pixel .style.posBottom: Float pointing und Integer</p>
.style.posHeight	<p>Objekthöhe identisch mit .style.height und .style.pixelHeight aber .style.height: String .style.pixelHeight: Integer, nur Pixel</p>



.style.posHeight: Float pointing und Integer

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
    document.all.tags("IMG").item(0).style.posHeight += 10;
</SCRIPT>
```

.style.position Art der Objektpositionierung innerhalb Eltern z.B. BODY

Beispiel 1:

```
<SPAN STYLE="position:relative; top:-3px">
    Testtext
</SPAN>
```

Beispiel 2:

```
<HEAD>
<STYLE>
    .StandardKlasse { position: static }
</STYLE>
<SCRIPT>
    function AbsolutPositionieren()
    {ID_Span.style.position="absolute";}

    function RelativPositionieren()
    {ID_Span.style.position="relative";}

    function HTMLPositionieren()
    {ID_Span.style.position="static";}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" CLASS="StandardKlasse">
        Testtext
    </SPAN>
    <INPUT onclick="RelativPositionieren()" TYPE=button VALUE="relativ">
    <INPUT onclick="AbsolutPositionieren()" TYPE=button VALUE="absolut">
    <INPUT onclick="HTMLPositionieren()" TYPE=button VALUE="HTML-statisch">
</BODY>
```

.style.posLeft Abstand linker Objektrand zur Umgebung rechte Kante
Abstand des linken Objektrandes des Kindes zum linken Objektrand des Elternelementes,
wenn im Elternelement STYLE="position:....." **kodiert wurde** (Wert von position ist egal)
Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):
Abstand der linken Kante des IMG zur linken Kante des DIV
identisch mit Attribut .style.left und .style.pixelLeft
aber .style.left: String
.style.pixelLeft: Integer, nur Pixel
.style.posLeft: Float pointing und Integer

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
    document.all.tags("IMG").item(0).style.posLeft -= 10;
</SCRIPT>
```

.style.posRight Abstand rechter Objektrand zur Umgebung linke Kante
identisch mit Attribut .style.rigth und .style.pixelRight
aber .style.right: String
.style.pixelRight: Integer, nur Pixel
.style.posRight: Float pointing und Integer

.style.posTop Abstand oberer Objektrand zur Umgebung Unterkante
Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes,
wenn im Elternelement STYLE="position:....." **kodiert wurde** (Wert von position ist egal)
Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):
Abstand der oberen Kante des IMG zur oberen Kante des DIV
identisch mit Attribut .style.top und .style.pixelTop
aber .style.top: String
.style.pixelTop: Integer, nur Pixel
.style.posTop: Float pointing und Integer

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
    document.all.tags("IMG").item(0).style.posTop -= 10;
</SCRIPT>
```

.style.posWidth Objektbreite
identisch mit .style.width und .style.pixelWidth



aber .style.width: String
 .style.pixelWidth: Integer, nur Pixel
 .style.posWidth: Float pointing und Integer

Beispiel:

```
<SCRIPT LANGUAGE="JScript">
  document.all.tags("IMG").item(0).style.posWidth += 10;
</SCRIPT>
```

.style.right

Abstand rechter Objektrand zur Umgebung linke Kante
 benötigt kodiertes .style.position Attribut
 identisch mit Attribut .style.posRigth und .style.pixelRight

aber .style.right: String
 .style.pixelRight: Integer, nur Pixel
 .style.posRight: Float pointing und Integer

Beispiel:

```
<DIV STYLE = "position:absolute; right:50px">
  ...
</DIV>
```

.style.rubyAlign

Ausrichtung des ruby Objektes
 hier nicht weiter erklärt
 im STYLE-Attribut ruby-align kodieren

.style.rubyOverhang

Überhang des ruby Objektes
 hier nicht weiter erklärt
 im STYLE-Attribut ruby-overhang kodieren

.style.rubyPosition

Position des ruby Objektes
 hier nicht weiter erklärt
 im STYLE-Attribut ruby-position kodieren

.style.scrollbar3dLightColor

xxxxxxxxxx	x	scrollbar3dLightColor
yyyyyyyyyyw	y	scrollbarHightlightColor
xy.....zW	scrollbarFaceColor
xy...../ \.....zW	/ \	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
xy.....zW		
xzzzzzzzzzW	z	scrollbarShadowColor
wwwwwwwww	w	scrollbarDarkShadowColor

Beispiel:

```
<HEAD>
<STYLE>
  TEXTAREA.Blue3dLight { scrollbar-3dlight-color:blue }
</STYLE>
</HEAD>
<BODY>
  <TEXTAREA CLASS="Blue3dLight">
    Testtext
  </TEXTAREA>
</BODY>
```

.style.scrollbarArrowColor

xxxxxxxxxx	x	scrollbar3dLightColor
yyyyyyyyyyw	y	scrollbarHightlightColor
xy.....zW	scrollbarFaceColor
xy...../ \.....zW	/ \	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
xy.....zW		
xzzzzzzzzzW	z	scrollbarShadowColor
wwwwwwwww	w	scrollbarDarkShadowColor

Beispiel:

```
<HEAD>
<STYLE>
  TEXTAREA.BlueArrow { scrollbar-arrow-color:blue }
</STYLE>
</HEAD>
<BODY>
  <TEXTAREA CLASS="BlueArrow">
    Testtext
  </TEXTAREA>
</BODY>
```

.style.scrollbarBaseColor

Basisfarbe aller Scrollbalken-Elemente
 automatische Farbstafflung für sämtliche Teilkomponentender Scrollbalken-Elemente

Beispiel 1:

```
<HEAD>
<STYLE>
```



```

TEXTAREA.BlueScrollbar { scrollbar-base-color:blue }
</STYLE>
</HEAD>
<BODY>
    <TEXTAREA CLASS="BlueScrollbar">
        Testtext
    </TEXTAREA>
</BODY>

```

Beispiel 2:

```

<HEAD>
<STYLE>
    BODY { scrollbar-base-color:darkolivegreen; background-color:tan }
    H1 { color:bisque }
    P { color:darkslategray }
</STYLE>
</HEAD>

```

.style.scrollbarDarkShadowColor

xxxxxxxxxxx	x	scrollbar3dLightColor
yyyyyyyyyyw	y	scrollbarHightlightColor
xy.....zW	scrollbarFaceColor
xy...../\.....zW	/\	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
xy.....zW		
xzzzzzzzzzW	z	scrollbarShadowColor
wwwwwwwww	w	scrollbarDarkShadowColor

Beispiel:

```

<HEAD>
<STYLE>
    TEXTAREA.BlueShadow { scrollbar-darkshadow-color:blue }
</STYLE>
</HEAD>
<BODY>
    <TEXTAREA CLASS="BlueShadow">
        Testtext
    </TEXTAREA>
</BODY>

```

.style.scrollbarFaceColor

xxxxxxxxxxx	x	scrollbar3dLightColor
yyyyyyyyyyw	y	scrollbarHightlightColor
xy.....zW	scrollbarFaceColor
xy...../\.....zW	/\	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
xy.....zW		
xzzzzzzzzzW	z	scrollbarShadowColor
wwwwwwwww	w	scrollbarDarkShadowColor

Beispiel:

```

<HEAD>
<STYLE>
    TEXTAREA.BlueFace { scrollbar-face-color:blue }
</STYLE>
</HEAD>
<BODY>
    <TEXTAREA CLASS="BlueFace">
        Testtext
    </TEXTAREA>
</BODY>

```

.style.scrollbarHighlightColor

xxxxxxxxxxx	x	scrollbar3dLightColor
yyyyyyyyyyw	y	scrollbarHightlightColor
xy.....zW	scrollbarFaceColor
xy...../\.....zW	/\	scrollbarArrowColor (nur falls Pfeil vorhanden ist)
xy.....zW		
xzzzzzzzzzW	z	scrollbarShadowColor
wwwwwwwww	w	scrollbarDarkShadowColor

Beispiel:

```

<HEAD>
<STYLE>
    TEXTAREA.BlueHighlight { scrollbar-highlight-color:blue }
</STYLE>
</HEAD>
<BODY>
    <TEXTAREA CLASS="BlueHighlight">
        Testtext
    </TEXTAREA>

```



</BODY>

.style.scrollbarShadowColor xxxxxxxxxx x scrollbar3dLightColor
 xxxxxxxxxxw y scrollbarHightlightColor
 xy.....zw scrollbarFaceColor
 xy...../ \zw / \ scrollbarArrowColor (nur falls Pfeil vorhanden ist)
 xy.....zw
 xxxxxxxxxxw z scrollbarShadowColor
 wwwwwwww w scrollbarDarkShadowColor

Beispiel:

```
<HEAD>
<STYLE>
  TEXTAREA.BlueShadow { scrollbar-shadow-color:blue }
</STYLE>
</HEAD>
<BODY>
  <TEXTAREA CLASS="BlueShadow">
    Testtext
  </TEXTAREA>
</BODY>
```

.style.scrollbarTrackColor Farbe des Trackelementes von Scrollbalken (Track = ziehen)

Beispiel:

```
<HEAD>
<STYLE>
  TEXTAREA.BlueTrack { scrollbar-track-color:blue }
</STYLE>
</HEAD>
<BODY>
  <TEXTAREA CLASS="BlueTrack">
    Testtext
  </TEXTAREA>
</BODY>
```

.style.styleFloat Textumfluss links, rechts
 unter IE 5.x: DIV und SPAN müssen .style.width kodiert haben
 siehe auch Eigenschaften .style.whiteSpace
 .style.clear
 .style.float

Beispiel:

```
<IMG ID="ID_IMG" SRC="test.jpg">
<BUTTON        onmouseover="ID_IMG.style.styleFloat='right'"
               onmouseout="ID_IMG.style.styleFloat='left'"
>
  Flip-Flop-Bild
</BUTTON>
```

.style.tableLayout Tabellen-Layout (auto oder fixed)
 siehe auch Eigenschaft .style.minHeight
 Tabelle mit fixiertem Layout erzeugen per Eigenschaft .style.tableLayout mit Wert "fixed"
 wird schneller dargestellt als bei Auto-Layout
 Hinweis: Standard bei Tabellen Eigenschaft .style.tableLayout mit Wert "auto"
 also Auto-Layout, also kein fixiertes Tabellenlayout

Beispiel:

```
<TABLE STYLE="table-layout:fixed" WIDTH=600>
  <COL WIDTH=100>
    <COL WIDTH=300>
      <COL WIDTH=200>
        <TR HEIGHT=20>
          <TD>...</TD>
          <TD>...</TD>
          <TD>...</TD>
        </TR>
      ....
    </TABLE>
```

.style.textAlign Textausrichtung nur für Block-Elemente wie SPAN oder DIV
 siehe auch .style.textAlignLast und .style.textJustify

Beispiel 1:

```
<HEAD>
<STYLE>
  P { text-align:center }
```



```

        .align1 { text-align:right }
        .align2 { text-align:justify }
    </STYLE>
</HEAD>
<BODY>
    <P      onclick= "this.className='align1'"
           ondblclick="this.className='align2'"
    >
        ...
    </P>
</BODY>

```

Beispiel 2:

```

<P      STYLE="font-size:14"
onmouseover="this.style.textAlign='center'"
>
    ...
</P>

```

.style.textAlignLast Textausrichtung der letzten Zeile
 nur für Block-Elemente wie SPAN oder DIV
 siehe auch **.style.textAlign** und **.style.textJustify**

Beispiel:

```

<P      STYLE="font-size:14"
onmouseover="this.style.textAlignLast='center'"
>
    ...
</P>

```

.style.textAutospace Zusatz-Textabstand bei asiatischen Zeichen

.style.textDecoration dekoratives Layout eines Textes für nichtleere Textobjekte (unterstreichen, überstreichen, durchstreichen)
 z.B. ist ein leeres
 bei Block-Element: Layout wird an Kinder vererbt
 siehe auch Eigenschaft **.style.textDecorationBlink**

Beispiel 1:

```

<DIV STYLE="text-decoration:line-through">
    .....
</DIV>

```

Beispiel 2:

```

<SPAN  STYLE="font-size:14"
onmouseover="this.style.textDecoration='underline'"
>
    .....
</SPAN>

```

.style.textDecorationBlink Wert der Eigenschaft **.style.textDecoration** auf "blink" prüfen
 Text auf blinkend prüfen
 true **.textDecoration** steht auf "blink"
 false **.textDecoration** steht auf "blink"

.style.textDecorationLineThrough Textdekoration "line-through" (durchstreichen)

Beispiel:

```

<P onclick="this.style.textDecorationLineThrough=true;">
    Testtext
</P>

```

.style.textDecorationNone Textdekoration "none" (deaktivieren aller aktiven Textdekorationen)
 true none aktiv, es wird deaktiviert
 false none nicht aktiv, kein Deaktivieren

.style.textDecorationOverline Textdekoration "overline" (überstreichen)

Beispiel:

```

<P      onmouseover="this.style.textDecorationOverline=true;">
    Testtext
</P>

```

.style.textDecorationUnderline Textdekoration "underline" (unterstreichen)
 siehe auch Eigenschaft **.style.textUnderlinePosition**

Beispiel:

```

<P onclick="this.style.textDecorationUnderline=true;">

```



Testtext
</P>

.style.textIndent Position-Ident in der der ERSTEN Zeile eines Textes
Ident nicht möglich in der Mitte eines umgebrochenen Objekts
(Umbrechung z.B. per BR)

Beispiel 1:

```
<HEAD>
<STYLE>
    DIV { text-indent:2cm }
    .click1 { text-indent:50% }
    .click2 { text-indent: }
</STYLE>
</HEAD>
<BODY>
<DIV onclick="this.className='click1'"
ondblclick="this.className='click2'"
>
    ...
</DIV>
```

Beispiel 2:

```
<DIV onmouseover="this.style.textIndent=2cm">
    .....
</DIV>
```

.style.textJustify Textausrichtung Blocksatz nur für Blockelemente wie SPAN oder DIV
Attribut **.style.textAlign** muss Wert "justify" haben

Beispiel:

```
<DIV STYLE="text-align='justify'; text-justify='auto'">
    .....
</DIV>
```

.textKashidaSpace Typographischer Effekt "Kashida" für Arabisch
siehe auch Eigenschaft **.style.textJustify**
nur für Blockelemente wie DIV und SPAN

.style.textOverflow Rahmen um Text
ab IE 6.x
Eigenschaft ermöglicht einen Rahmen um einen Text, wobei der Text im Rahmen überlaufen kann
Überlauf und Rahmen können folgende Formen haben:
Abschneiden an Rahmengrenze
Abschneiden an Rahmengrenze mit automatischem Anfügen von ".." als Andeutung einer Fortsetzung
kein Abschneiden, also Rahmen in voller Textbreite
Achtung: Damit der Text nicht umgebrochen wird im Rahmen, muss die Eigenschaft **.style.whiteSpace** auf "nowrap" gesetzt sein
bzw. der Text in das NOBR-Tag eingeschlossen sein.
Desweiteren sollte die Eigenschaft **.style.overflow** auf "hidden" gesetzt sein, damit keine Scrollbalken erscheinen (im Gegensatz zu "auto" oder "scroll")

Beispiel 1 Rahmen mit Abschneiden an Rahmengrenze

```
STYLE="text-overflow : clip; overflow : hidden"
```

Beispiel 2 Rahmen mit Abschneiden aber automatisch ".." als Andeutung der Fortsetzung

```
STYLE="text-overflow : ellipsis; overflow : hidden"
```

Beispiel 3 Rahmen in kompletter Breite des Textes

```
STYLE="text-overflow : ellipsis; overflow : visible"
```

.style.textTransform Textkonvertierung nach Gross oder Klein oder erstes Zeichen im Wort stets groß
siehe auch Objekt String

Beispiel 1:

```
<HEAD>
<STYLE>
    .transform1 { text-transform:uppercase }
    .transform2 { text-transform:lowercase }
    .transform3 { text-transform:none }
</STYLE>
```




```

</HEAD>
<BODY>
    <DIV    STYLE="font-size:14"
           onmouseover="this.className='transform1'"
           onclick="this.className='transform2'"
           ondblclick="this.className='transform3'"
    >
        ....
    </DIV>
</BODY>

```

Beispiel 2:

```

<DIV    STYLE="font-size:14"
           onmouseover="this.style.textTransform='uppercase'"
           onmouseout="this.style.textTransform='lowercase'"
           onclick="this.style.textTransform='none'"
    >
        .....
</DIV>

```

`.style.textUnderlinePosition` Position der Textdekoration "underline" (unterstreichen)
 siehe auch Eigenschaft `.style.textDecorationUnderline`

Beispiel 1:

```

<HTML STYLE="text-underline-position:above">
<BODY>
<SPAN STYLE="text-decoration:underline">
    Testtext
</SPAN>
</BODY>
</HTML>

```

`.style.top` Abstand oberer Objektrand zur Umgebung Unterkante
 Abstand des oberen Objektrandes des Kindes zum oberen Objektrand des Elternelementes,
wenn im Elternelement `STYLE="position:....."` **kodiert wurde** (Wert von position ist egal)
 Bsp.: IMG als Kind eines DIV (IMG liegt innerhalb eines DIV):
 Abstand der oberen Kante des IMG zur oberen Kante des DIV
 benötigt kodierte Eigenschaft `.style.position`
 identisch mit Attribut `.style.posTop` und `style.pixelTop`
 aber `.style.top`: String
 `.style.pixelTop`: Integer, nur Pixel
 `.style.posTop`: Float pointing und Integer

Beispiel 1:

```

<DIV STYLE="position:absolute;top:100px">
    ...
</DIV>

```

Beispiel 2:

```

<IMG    SRC="test.jpg"
        STYLE="position:absolute; top:80px;" onmouseover="this.style.top='100px'"
        onmouseout="this.style.top='80px'"
    >

```

`.style.unicodeBidi` Bidirektionale Dateneingabe von Unicode in das Element (Zeichenstrom, der vom Elternobjekt
 in das Element fließt, kann seine eigene Flussrichtung besitzen).

Typische Anwendung ist z.B. der Mix aus Zeichen, die mal von links nach rechts und mal von
 rechts nach links gerendert werden sollen. Damit ist für den User eine variable Leserichtung
 möglich, falls der User beide Richtungen benutzen will.

Es ist möglich, den Datenstrom von Zeichen trotz aktiver bidirektionaler Dateneingabe
 genau nach Vorgabe laut Eigenschaft `.style.direction` fließen zu lassen.
 Bsp.: Wenn `.style.direction` auf "ltr" gesetzt ist, so wird diese Richtung konsequent eingehalten,
 auch wenn der Datenstrom von rechts nach links fließen will.

Die Eigenschaft `.style.direction` sollte auf "inherit" gesetzt sein, wenn bidirektionaler Datenfluss
 immer erfolgreich sein soll und die Flussrichtung des Datenstromes unbekannt ist.
 Damit wird die Richtung des Elternobjektes übernommen.

`.style.verticalAlign` Ausrichtung vertikal für Objekte
 mit VALIGN-Attribut
 und bei Textobjekten ohne VALIGN-Attribut

Beispiel:

```

<TABLE BORDER width=100>
<TR>
    <TD    onmouseover="this.style.verticalAlign='bottom'"

```



```

                                onmouseout="this.style.verticalAlign=""
                                >
                                Testtext
                                </TD>
                            </TR>
                        </TABLE>

```

.style.visibility Objekt-Sichtbarkeit ohne Reservierung des Platzs im Layout der Umgebung
 ab IE 5.x Sichtbarkeit eines Kindobjektes, auch wenn Eltern unsichtbar sind
 siehe auch Eigenschaften

```

        .style.overflow
        .style.overflowX
        .style.overflowY
        .style.clip
        .style.display

```

Hinweis: display:none

Objekt behält den Platz im Layout der Umgebung,
 obwohl Objekt unsichtbar ist

Beispiel 1:

```

<HEAD>
<STYLE>
    .vis1 { visibility:visible }
    .vis2 { visibility:hidden }
</STYLE>
</HEAD>
<BODY>
    <IMG ID="ID_IMG" SRC="test.jpg">
    <P      onmouseover="ID_IMG.className='vis2'"
           onmouseout="ID_IMG.className='vis1'"
    >
        Testtext
    </P>
</BODY>

```

Beispiel 2:

```

<SCRIPT>
    function Verstecken()
    {ID_IMG.style.visibility="hidden"; }

    function Anzeigen()
    {ID_IMG.style.visibility="visible"; }
</SCRIPT>
<IMG ID="ID_IMG"
SRC="test.jpeg">
<SPAN onmouseover="Verstecken()"
onmouseout="Anzeigen()"
>
    Testtext
</SPAN>

```

.style.whiteSpace automatischer Zeilenumbruch bei Block-Elementen wie DIV und SPAN
 Umbrechen erfolgt an den Stellen vo Blanks, Tabs stehen, aber nicht an der Stelle
 eines geschützten Blanks ();
 siehe auch Eigenschaften

```

        .style.pageBreakAfter
        .style.pageBreakBefore
        .style.wordBreak
        .style.wordWrap

```

Beispiel:

```

<HEAD>
<STYLE>
    .clsOneLiner { white-space: nowrap}
    .clsAutoBreak { white-space: normal}
</STYLE>
</HEAD>
<BODY>
<P      onmouseover="this.className='clsOneLiner';"
           onmouseout="this.className='clsAutoBreak';"
    >
        Testtext
    </P>
</BODY>

```

.style.width Objektbreite
 identisch mit Attribut .style.posWidth und .style.pixelWidth
 aber .style.width: String
 .style.pixelWidth: Integer, nur Pixel



.style.posWidth: Float pointing und Integer

Beispiel 1:

```
<DIV STYLE="position:absolute;top:10px;left:10px;width=1in">
.....
</DIV>
```

Beispiel 2:

```
<IMG SRC="test.jpg"
onclick="this.style.width='1cm'"
ondblclick="this.style.width=""
>
```

.style.wordBreak

Zeilenumbruch in Worten
siehe auch Eigenschaften

.style.pageBreakAfter
.style.pageBreakBefore
.style.wordWrap
.style.whiteSpace

Beispiel:

```
<DIV STYLE="word-break=normal">
.....
</DIV>
```

.style.wordSpacing

Zusätzlicher Platz zwischen Worten
ab IE 6.x

Beispiel:

```
<HEAD>
<STYLE>
SPAN.spacing{ word-spacing: 10;}
</STYLE>
<SCRIPT>
function PlatzAendern()
{
    ID_Span.style.wordSpacing =
    ID_Select.options[ID_Select.selectedIndex].text;
}
</SCRIPT>
<SELECT ID = "ID_Select" onchange = "PlatzAendern()">
<OPTION>10
<OPTION>15
<OPTION>20
</SELECT>
<SPAN ID = "ID_Span" CLASS = "spacing">
Testtext
</SPAN>
```

.style.wordWrap

Wortumbruch bei Überschreitung der Objektgrenzen
für Block-Elemente wie DIV oder SPAN
Elemente müssen kodiert haben

.style.height und/oder .style.width
.style.position mit "absolute"

siehe auch Eigenschaften

.style.pageBreakAfter
.style.pageBreakBefore
.style.wordBreak
.style.whiteSpace

Beispiel:

```
<P STYLE="word-wrap:break-word;width:100%;left:0">
Testtext
</P>
```

.style.writingMode

Flussrichtung bei Objektanzeige
wird nicht vererbt
ersetzt Eigenschaft .style.layoutFlow, da diese deprecated ist
siehe auch Eigenschaft .style.verticalAlign

Beispiel:

```
<HEAD>
<STYLE>
.clsHoriz { writing-mode:lr-tb }
</STYLE>
</HEAD>
<BODY>
<DIV STYLE="writing-mode:tb-rl">
vertikal
<SPAN CLASS="clsHoriz">
horizontal
```



```

        </SPAN>
        vertikal
        <SPAN STYLE="writing-mode:lr-tb">
            horizontal
        </SPAN>
    </DIV>
</BODY>

```

.style.zIndex Reihenfolge von überlappten Objekten
 nur für Elemente mit kodierter Eigenschaft `.style.position` mit dem Wert "relative" oder "absolute"
 nicht für Elemente in oder mit Fenster
 z.B. Control-Objekte mit eigenem Fenster wie Dialogbox
 im Fenster selektieries Objekt
 ab IE 5.5: iframe Objekt unterstützt z-index, da fensterlos
 siehe auch layer Objekt des Netscape unter 6.x

Beispiel 1:

```

<IMG SRC="test.jpg" STYLE="position:absolute; top:100; left:100; z-index:4">
<DIV STYLE="position:absolute; top:100; left:100; color:red; background-color:beige; font-weight:bold; z-index:1">
    .....
</DIV>

```

Beispiel 2:

```

<IMG ID="ID_Img1" SRC="test1.jpeg"
STYLE="position:absolute;top:10px;left:10px;"
onclick="ID_Img1.style.zIndex=1; ID_Img2.style.zIndex=2"
>
<IMG ID="ID_Img2" SRC="test2.jpg"
STYLE="position:absolute;top:1px;left:1px;"
onclick="ID_Img1.style.zIndex=2; ID_Img2.style.zIndex=1"
>

```

.style.zoom Objekt zoomen mit Layoutkorrektur der Objektumgebung
 Hinweis: Ein Fenster kann nur nach rechts und unten erweitert werden, wenn das gezoomte Objekt die Fenstergrenze überschreitet (falls das Fenster überhaupt erweiterbar ist). Das Zoomen wird also als Erweiterung gegenüber der linken oberen Ecke des Objektes stattfinden.
 Als Zoombasis dient immer die Objektgröße **vor dem ersten** Zoomen und nicht die aus dem Zoomvorgang resultierende (kumulative) Objektgröße.

Beispiel 1:

```

<HEAD>
<STYLE>
.clsTeenyWeeny { zoom: 0.10 }
</STYLE>
</HEAD>

```

Beispiel 2:

```

<P onmouseover="this.style.zoom='200%'"
onmouseout="this.style.zoom='normal'"
>
    Testtext
</P>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    var ZoomFaktorStartWert = 10;           // > 0, ganzzahlig, in Prozent
    var ZoomSchrittWeite = 1;               // > 0, ganzzahlig
    var DrehSchrittWeite = 1;               // 1 oder 2 oder 3 da Faktor von 120 Grad
    var ZoomFaktor = ZoomFaktorStartWert;   // > 0, ganzzahlig, in Prozent

    function VergroessernVerkleinern()
    {
        if (event.wheelDelta >= (DrehSchrittWeite * 120))
        { ZoomFaktor += ZoomSchrittWeite; }
        else
        {
            if (event.wheelDelta <= (-1 * (DrehSchrittWeite * 120)))
            { ZoomFaktor -= ZoomSchrittWeite; }
        }

        if (ZoomFaktor <= 0)
        { ZooFaktor = ZoomFaktorStartWert; }

        ID_Image.style.zoom = ZoomFaktor + '%';
    }

```



```

    }
</SCRIPT>
</HEAD>
<BODY>
    <IMG ID="ID_Image" SRC="test.jpg" onmousewheel="VergroessernVerkleinern()">
</BODY>
</HTML>

```

Beispiel 4:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
    var BildUrl="zoom.jpg";
    var BildBreite=250;
    var BildHoehe=372;

    // Bildobjekte vorladen
    // Image-Zeiger
    BildZeiger = new Image();

    // Url dem Zeiger zuweisen und Bild somit vorladen
    BildZeiger.src= BildUrl;

    // .style.zoom
    //          ändert leider die DIV-Dimensionen, so dass die Input-Button VOR dem
    //          zu zommenden Bild liegen müssen und somit ein DIV nicht nötig wird
    //          kann nicht mit einem Zeichenkettenwert belegt werden, sondern muss direkt kodiert
    //          werden !
    document.write(
        '<FORM NAME="Formular">'
        + '<INPUT NAME="Button20" TYPE=button VALUE="20%" onclick="Bild_ID.style.zoom=\'20%\''
        + '<INPUT NAME="Button40" TYPE=button VALUE="40%" onclick="Bild_ID.style.zoom=\'40%\''
        + '<INPUT NAME="Button60" TYPE=button VALUE="60%" onclick="Bild_ID.style.zoom=\'60%\''
        + '<INPUT NAME="Button80" TYPE=button VALUE="80%" onclick="Bild_ID.style.zoom=\'80%\''
        + '<INPUT NAME="Button100" TYPE=button VALUE="100%" onclick="Bild_ID.style.zoom=\'100%\''
        + '<INPUT NAME="Button120" TYPE=button VALUE="120%" onclick="Bild_ID.style.zoom=\'120%\''
        + '<INPUT NAME="Button140" TYPE=button VALUE="140%" onclick="Bild_ID.style.zoom=\'140%\''
        + '<INPUT NAME="Button160" TYPE=button VALUE="160%" onclick="Bild_ID.style.zoom=\'160%\''
        + '<INPUT NAME="Button180" TYPE=button VALUE="180%" onclick="Bild_ID.style.zoom=\'180%\''
        + '<INPUT NAME="Button200" TYPE=button VALUE="200%" onclick="Bild_ID.style.zoom=\'200%\''
        + '<INPUT NAME="Button300" TYPE=button VALUE="300%" onclick="Bild_ID.style.zoom=\'300%\''
        + '<INPUT NAME="Button400" TYPE=button VALUE="400%" onclick="Bild_ID.style.zoom=\'400%\''
        + '</FROM>\n'
        + '<BR>\n'
        );

    document.write(
        '<IMG ID="Bild_ID" SRC="" + BildZeiger.src + "" '
        + 'HEIGHT=' + BildHoehe
        + ' WIDTH=' + BildBreite
        + '>\n');

    document.all.Bild_ID.style.zoom='100%'; // .style ist eine document.all-Eigenschaft
                                           // Bild_ID ist Objekt von document
                                           // entspricht document.Formular.Button100.click();

    document.Formular.Button100.focus(); // entspricht document.Formular.elements[4].focus();
                                           // funktioniert leider nur, wenn Dokument nicht aus IE-Cache
                                           // gelesen wird (sonst Übernahme der letzten
                                           // Fokuspotion)

// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

7.3. Style-Methoden

url() Diese Methode ist nicht direkt als Methode des Style-Objektes implementiert und darf daher **nicht** mit Punktnotation kodiert werden (vermutlich parserintern)

Beispiele für Bild:

```

<STYLE>
    .style1 {background:beige url(sphere.jpg) no-repeat top center}

```



```

        .style2{ background:ivory url(sphere.jpeg) no-repeat bottom right }
    </STYLE>

    <SPAN onclick="this.style.background='beige url(sphere.jpeg) no-repeat top center'"></SPAN>

    <STYLE>
        .setUrl { background-image: url(sphere.jpg) }
        .loseUrl { background-image: url(none) }
    </STYLE>
    <SPAN STYLE="font-size:14"
        onmouseover="this.className='setUrl'"
        onmouseout="this.className='loseUrl'"
    >
    </SPAN>

```

Beispiele für Behavior:

```

url(#objID)           mit objID als ID des OBJECT-Tags
url(#default#behaviorName)  eines Standard-Behaviors des IE

STYLE="behavior:url(a1.htc) url(a2.htc)"

<STYLE>
    .Klasse{ behavior:url(#myObject) }
</STYLE>

<STYLE>
    DIV { behavior:url(fly.htc) url (zoom.htc) url (fade.htc)}
</STYLE>

```

Beispiel für Cursorformen aus **Datei** und **nicht** für im Browser implementierte Standard-Cursorformen:
url('mycursor.cur')

Hinweise: alert (zeiger_auf_objekt.style.cursor); liefert beim IE den String 'url(cursor_form)'
mit cursor_form für den aktuellen **und** gesetzten Cursor.

style.cursor ist standardgemäß mit einer Leerkette belegt, solange **kein** Cursor gesetzt wird
kann nicht mit url(cursor_form) belegt werden, wenn es sich um eine **standardgemäß im Browser implementierte** Cursorform handelt wie z.B. 'hand' oder 'normal'.
Grund: Diese Cursorformen sind keine Dateien, obwohl alert den String 'url(cursor_form)' anzeigt.



8. Anhang: Events des Internet Explorer

Hinweis: Events aller Objekte im window werden per gemeinsamen window.event verwaltet.

window.event ist nur dann verfügbar, wenn ein Event abgefangen wird (z.B. per Eventhandler), denn es muss das Objekt, welches ein Event ausgelöst hat, ermittelbar sein.

.fireEvent() kann nur von einem Event-Handler aus aktiviert werden, wobei die Besonderheiten von fireEvent() zu beachten sind.

Eventhandler attachieren kann erfolgen per

HTML-Code per onxxx="..."

attachEvent()

createElement(html_tag_mit_begrenzer_und_mit_onxxx="....")

Scriptanweisung object.onxxx=evenhandler_funktions_bezeichner_ohne_klammern;

IE-eigener SCRIPT-Tag-Kodierung für Events

Events können entlang der Objekthierarchie weitergereicht werden (wie im Wasser aufsteigende Blasen .. bubbles)

Events könnten eventuell gecancelt werden.

8.1. wichtige Objekte und deren Events(Auswahl) - Übersicht

a Objekt

onactivate	ondragend	onmousemove
onafterupdate	ondragenter	onmouseout
onbeforeactivate	ondragleave	onmouseover
onbeforecopy	ondragover	onmouseup
onbeforecut	ondragstart	onmousewheel
onbeforedeactivate	ondrop	onmove
onbeforeeditfocus	onerrorupdate	onmoveend
onbeforepaste	onfocus	onmovestart
onbeforeupdate	onfocusin	onpaste
onblur	onfocusout	onpropertychange
onclick	onhelp	onreadystatechange
oncontextmenu	onkeydown	onresize
oncontrolselect	onkeypress	onresizeend
oncopy	onkeyup	onresizestart
oncut	onlosecapture	onselectstart
ondblclick	onmousedown	ontimeerror
ondeactivate	onmouseenter	
ondrag	onmouseleave	

applet Objekt

onactivate	ondeactivate	onmouseup
onbeforeactivate	onfocus	onmousewheel
onbeforecut	onfocusin	onmove
onbeforedeactivate	onfocusout	onmoveend
onbeforeeditfocus	onhelp	onmovestart
onbeforepaste	onkeydown	onpaste
onblur	onkeypress	onpropertychange
oncellchange	onkeyup	onreadystatechange
onclick	onload	onresize
oncontextmenu	onlosecapture	onresizeend
oncontrolselect	onmousedown	onresizestart
oncut	onmouseenter	onrowenter
ondataavailable	onmouseleave	onrowexit
ondatasetchanged	onmousemove	onrowsdelete
ondatasetcomplete	onmouseout	onrowsinserted
ondblclick	onmouseover	onscroll

area Objekt

onactivate	ondrag	onmousedown
onbeforeactivate	ondragend	onmouseenter
onbeforecopy	ondragenter	onmouseleave
onbeforecut	ondragleave	onmousemove
onbeforedeactivate	ondragover	onmouseout
onbeforeeditfocus	ondragstart	onmouseover
onbeforepaste	ondrop	onmouseup
onblur	onfocus	onmousewheel
onclick	onfocusin	onmove
oncontextmenu	onfocusout	onmoveend
oncontrolselect	onhelp	onmovestart
oncopy	onkeydown	onpaste
oncut	onkeypress	onpropertychange
ondblclick	onkeyup	onreadystatechange
ondeactivate	onlosecapture	onresizeend



onresizestart

onselectstart

ontimeerror

bgsound Objekt

onlayoutcomplete

onmouseenter

onmouseleave

onreadystatechange

body Objekt

onactivate

onafterprint

onbeforeactivate

onbeforecut

onbeforedeactivate

onbeforeeditfocus

onbeforepaste

onbeforeprint

onbeforeunload

onclick

oncontextmenu

oncontrolselect

oncut

ondblclick

ondeactivate

ondrag

ondragend

ondragenter

ondragleave

ondragover

ondragstart

ondrop

onfilterchange

onfocusin

onfocusout

onkeydown

onkeypress

onkeyup

onload

onlosecapture

onmousedown

onmouseenter

onmouseleave

onmousemove

onmouseout

onmouseover

onmouseup

onmousewheel

onmove

onmoveend

onmovestart

onpaste

onpropertychange

onreadystatechange

onresizeend

onresizestart

onscroll

onselect

onselectstart

onunload

button Objekt

onactivate

onafterupdate

onbeforeactivate

onbeforecut

onbeforedeactivate

onbeforeeditfocus

onbeforepaste

onbeforeupdate

onblur

onclick

oncontextmenu

oncontrolselect

oncut

ondblclick

ondeactivate

ondragenter

ondragleave

ondragover

ondrop

onerrorupdate

onfilterchange

onfocus

onfocusin

onfocusout

onhelp

onkeydown

onkeypress

onkeyup

onlosecapture

onmousedown

onmouseenter

onmouseleave

onmousemove

onmouseout

onmouseover

onmouseup

onmousewheel

onmove

onmoveend

onmovestart

onpaste

onpropertychange

onreadystatechange

onresize

onresizeend

onresizestart

onselectstart

ontimeerror

comment Objekt

onpropertychange

onreadystatechange

custom Objekt

onactivate

onafterupdate

onbeforeactivate

onbeforecopy

onbeforecut

onbeforedeactivate

onbeforeeditfocus

onbeforepaste

onbeforeupdate

onblur

onclick

oncontextmenu

oncontrolselect

oncopy

oncut

ondblclick

ondeactivate

ondrag

ondragend

ondragenter

ondragleave

ondragover

ondragstart

ondrop Fires

onerrorupdate

onfilterchange

onfocus

onfocusin

onfocusout

onhelp

onkeydown

onkeypress

onkeyup

onlosecapture

onmousedown

onmouseenter

onmouseleave

onmousemove

onmouseout

onmouseover

onmouseup

onmousewheel

onmove

onmoveend

onmovestart

onpaste

onpropertychange

onreadystatechange

onresize

onresizeend

onresizestart

onscroll

onselectstart

div Objekt

onactivate

onafterupdate

onbeforeactivate

onbeforecopy

onbeforecut

onbeforedeactivate

onbeforeeditfocus

onbeforepaste Fires

onbeforeupdate

onblur

onclick

oncontextmenu

oncontrolselect

oncopy

oncut

ondblclick

ondeactivate

ondrag

ondragend

ondragenter

ondragleave



ondragover	onlayoutcomplete
ondragstart	onlosecapture
ondrop	onmousedown
onerrorupdate	onmouseenter
onfilterchange	onmouseleave
onfocus	onmousemove
onfocusin	onmouseout
onfocusout	onmouseover
onhelp	onmouseup
onkeydown	onmousewheel
onkeypress	onmove
onkeyup	onmoveend

document Objekt

onactivate	ondragenter
onbeforeactivate	ondragleave
onbeforecut	ondragover
onbeforedeactivate	ondragstart
onbeforeeditfocus	ondrop
onbeforepaste	onfocusin
onclick	onfocusout
oncontextmenu	onhelp
oncontrolselect	onkeydown
oncut	onkeypress
ondblclick	onkeyup
ondeactivate	onmousedown
ondrag	onmousemove
ondragend	onmouseout

embed Objekt

onactivate	onfocus
onbeforeactivate	onfocusin
onbeforecut	onfocusout
onbeforedeactivate	onhelp
onbeforepaste	onload
onblur	onlosecapture
onclick	onmousedown
oncontextmenu	onmouseenter
oncontrolselect	onmouseleave
oncut	onmousemove
ondblclick	onmouseout
ondeactivate	onmouseover

fieldset Objekt

onactivate	ondragenter
onbeforeactivate	ondragleave
onbeforecopy	ondragover
onbeforecut	ondragstart
onbeforedeactivate	ondrop
onbeforeeditfocus	onfilterchange
onbeforepaste	onfocus
onblur	onfocusin
onclick	onfocusout
oncontextmenu	onhelp
oncontrolselect	onkeydown
oncopy	onkeypress
oncut	onkeyup
ondblclick	onlosecapture
ondeactivate	onmousedown
ondrag	onmouseenter
ondragend	onmouseleave

font Objekt

onactivate	ondblclick
onbeforeactivate	ondeactivate
onbeforecut	ondrag
onbeforedeactivate	ondragend
onbeforeeditfocus	ondragenter
onbeforepaste	ondragleave
onblur	ondragover
onclick	ondragstart
oncontextmenu	ondrop
oncontrolselect	onfocus
oncut	onfocusin

onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart
ontimeerror

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectionchange
onstop

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlayoutcomplete
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove



onmouseout
onmouseover
onmouseup
onmousewheel
onmove

onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange

onresizeend
onresizestart
onselectstart
ontimeerror

form Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onreset
onresize
onresizeend
onresizestart
onselectstart
onsubmit
ontimeerror

frame Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeupdate
onblur
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onload
onmove
onmoveend

onmovestart
onresize
onresizeend
onresizestart

frameset Objekt

onactivate
onafterprint
onbeforedeactivate
onbeforeprint
onbeforeunload
onblur

oncontrolselect
ondeactivate
onfocus
onload
onmove
onmoveend

onmovestart
onresizeend
onresizestart
onunload

head Objekt

onlayoutcomplete

onreadystatechange

html Objekt

onlayoutcomplete
onmouseenter

onmouseleave
onreadystatechange

html comment Objekt

onlayoutcomplete

iframe Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeupdate
onblur
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onload
onmove
onmoveend

onmovestart
onreadystatechange
onresizeend
onresizestart
ontimeerror

img Objekt

onabort
onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncopy

oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerror
onerrorupdate
onfilterchange
onfocus

onfocusin
onfocusout
onhelp
onload
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove



onmoveend
onmovestart
onpaste
onpropertychange

onreadystatechange
onresize
onresizeend
onresizestart

onselectstart
ontimeerror

input Objekt

keine Events

input button Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input checkbox Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

input file Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input hidden Objekt

onactivate
onafterupdate
onbeforedeactivate
onbeforeeditfocus
onbeforeupdate
oncontrolselect

ondeactivate
onerrorupdate
onfocus
onlosecapture
onmove
onmoveend

onmovestart
onpropertychange
onreadystatechange
onresizeend
onresizestart
ontimeerror

input image Objekt

onactivate
onbeforeactivate

onbeforecut
onbeforedeactivate

onbeforeeditfocus
onbeforepaste



onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop

onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input password Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input radio Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

input reset Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

input submit Objekt

onactivate
onbeforeactivate

onbeforecut
onbeforedeactivate

onbeforeeditfocus
onbeforepaste



onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
ontimeerror

onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart

input text Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onchange
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselect
onselectstart
ontimeerror

label Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart

link Objekt

onload

onreadystatechange

map Objekt

onbeforeactivate
onbeforecut
onbeforepaste
onclick
oncut
ondblclick
ondrag
ondragend
ondragenter
ondragleave
ondragover

ondragstart
ondrop
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onpaste
onpropertychange
onreadystatechange
onscroll
onselectstart

marquee Objekt

onactivate
onafterupdate

onbeforeactivate
onbeforecut

onbeforedeactivate
onbeforeeditfocus



onbeforepaste
onbeforeupdate
onblur
onbounce
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop

onerrorupdate
onfilterchange
onfinish
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart
onstart
ontimeerror

namespace Objekt

onreadystatechange

noframe Objekt

onreadystatechange

noscript Objekt

onreadystatechange

object Objekt

onactivate
onbeforedeactivate
onbeforeeditfocus
onblur
oncellchange
onclick
oncontrolselect
ondataavailable
ondatasetchanged
ondatasetcomplete
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerror
onfocus
onkeydown
onkeypress
onkeyup
onlosecapture
onmove

onmoveend
onmovestart
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onrowenter
onrowexit
onrowsdelete
onrowsinserted
onscroll
onselectstart

option Objekt

onlayoutcomplete
onlosecapture

onpropertychange
onreadystatechange

onselectstart
ontimeerror

popup Objekt

keine Events

scriptObjekt

onload

onpropertychange

onreadystatechange

select Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onchange
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate

ondragenter
ondragleave
ondragover
ondrop
onerrorupdate
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart

selection Objekt

ontimeerror

span Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag

ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter

onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

style Objekt

onerror

onreadystatechange

table Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave

ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselectstart
ontimeerror

table.caption Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.col Objekt

keine

table.colGroup Objekt

onreadystatechange

table.tBody Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu

oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter
ondragleave

ondragover
ondragstart
ondrop
onfocus
onfocusin
onfocusout
onhelp
onkeydown



onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart

onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tFoot Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondragenter
ondragstart

onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tHead Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondragenter
ondragstart

onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover

onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tr Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave

onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

table.tr.td Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick

ondeactivate
ondrag
ondragend
ondragenter
ondragleave
ondragover
ondragstart
ondrop
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown

onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart



onpaste
onpropertychange
onreadystatechange

onresizeend
onresizestart
onselectstart

ontimeerror

table.tr.th Objekt

onactivate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncopy
oncut
ondblclick
ondeactivate
ondragenter

ondragstart
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove
onmouseout

onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresizeend
onresizestart
onselectstart
ontimeerror

textarea Objekt

onactivate
onafterupdate
onbeforeactivate
onbeforecopy
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onbeforeupdate
onblur
onchange
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend

ondragenter
ondragleave
ondragover
ondragstart
ondrop
onerrorupdate
onfilterchange
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onscroll
onselect
onselectstart
ontimeerror

textrange Objekt

keine Events

var Objekt

onactivate
onbeforeactivate
onbeforecut
onbeforedeactivate
onbeforeeditfocus
onbeforepaste
onblur
onclick
oncontextmenu
oncontrolselect
oncut
ondblclick
ondeactivate
ondrag
ondragend
ondragenter

ondragleave
ondragover
ondragstart
ondrop
onfocus
onfocusin
onfocusout
onhelp
onkeydown
onkeypress
onkeyup
onlosecapture
onmousedown
onmouseenter
onmouseleave
onmousemove

onmouseout
onmouseover
onmouseup
onmousewheel
onmove
onmoveend
onmovestart
onpaste
onpropertychange
onreadystatechange
onresize
onresizeend
onresizestart
onselectstart
ontimeerror

window Objekt

onactivate
onafterprint
onbeforedeactivate
onbeforeprint
onbeforeunload
onblur
oncontrolselect

ondeactivate
onerror
onfocus
onhelp
onload
onmove
onmoveend

onmovestart
onresize
onresizeend
onresizestart
onscroll
onunload

xml Objekt

ondataavailable

ondatachanged

ondatasetcomplete



onreadystatechange
onrowenter

onrowexit
onrowsdelete

onrowsinserted

8.2. wichtige Events und deren Auftreten in Objekten - Übersicht

Hinweis: bubbles entspricht durchreichen der Events nach oben entlang der DOM-Hiearchie
cancels entspricht unterbrechbar

.fireEvent()

Fires a specified event on the object.

kann nur innerhalb eines Eventhandlers aktiviert werden, da ansonsten wirkungslos

Syntax

bFired = object.fireEvent(sEvent [, oEventObject])

sEvent Required. String that specifies the name of the event to fire.

oEventObject Optional. Object that specifies the event object from which to obtain event object properties.

Return Value

Boolean. Returns one of the following values:

true Event fired successfully.

false Event was cancelled.

fireEvent() setzt automatisch window.event.cancelBubble auf false, also kein Hochreichen des Events in der DOM-Hierarchie

fireEvent() liefert

true , wenn

Event erfolgreich gefeuert

oder Event nicht feuerebar aber auch nicht cancelbar

false, wenn

Event nicht erfolgreich gefeuert

UND nicht gecancelt wurde

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT>
function fnFireEvents()
{
    oDiv.innerText = "The cursor has moved over me!";
    oButton.fireEvent("onclick");
}
</SCRIPT>
</HEAD>
<BODY>
<h1>Using the fireEvent method</h1>
By moving the cursor over the DIV below, the button is clicked.
<P>
<DIV ID="oDiv" onmouseover="fnFireEvents();">
Mouse over this!
</DIV>
<p>
<BUTTON ID="oButton" ONCLICK="this.innerText='I have been clicked!'" >
Button</BUTTON>
</BODY>
</HTML>
```

Objekte mit der Methode sind

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, styleSheet, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, WBR, XML, XMP

window-Events

onactivate Fires when the object is set as the active element.

onafterprint Fires on the object immediately after its associated document prints or previews for printing.

onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.



onbeforeprint Fires on the object before its associated document prints or previews for printing.

onbeforeunload Fires prior to a page being unloaded.

onblur Fires when the object loses the input focus.

oncontrolselect Fires when the user is about to make a control selection of the object.

ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.

onerror Fires when an error occurs during object loading.

onfocus Fires when the object receives focus.

onhelp Fires when the user presses the F1 key while the browser is the active window.

onload Fires immediately after the browser loads the object.

onmove Fires when the object moves.

onmoveend Fires when the object stops moving.

onmovestart Fires when the object starts to move.

onresize Fires when the size of the object is about to change.

onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.

onresizestart Fires when the user begins to change the dimensions of the object in a control selection.

onscroll Fires when the user repositions the scroll box in the scroll bar on the object.

onunload Fires immediately before the object is unloaded.

document-Events

onactivate Fires when the object is set as the active element.

onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecut Fires on the source object before the selection is deleted from the document.

onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.

onbeforeeditfocus Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.

onbeforepaste Fires on the target object before the selection is pasted from the system clipboard to the document.

onclick Fires when the user clicks the left mouse button on the object.

oncontextmenu Fires when the user clicks the right mouse button in the client area, opening the context menu.

oncontrolselect Fires when the user is about to make a control selection of the object.

oncut Fires on the source element when the object or selection is removed from the document and added to the system clipboard.

ondblclick Fires when the user double-clicks the object.

ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.

ondrag Fires on the source object continuously during a drag operation.

ondragend Fires on the source object when the user releases the mouse at the close of a drag operation.

ondragenter Fires on the target element when the user drags the object to a valid drop target.

ondragleave Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.

ondragover Fires on the target element continuously while the user drags the object over a valid drop target.

ondragstart Fires on the source object when the user starts to drag a text selection or selected object.

ondrop Fires on the target object when the mouse button is released during a drag-and-drop operation.

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onhelp Fires when the user presses the F1 key while the browser is the active window.

onkeydown Fires when the user presses a key.

onkeypress Fires when the user presses an alphanumeric key.

onkeyup Fires when the user releases a key.

onmousedown Fires when the user clicks the object with either mouse button.

onmousemove Fires when the user moves the mouse over the object.

onmouseout Fires when the user moves the mouse pointer outside the boundaries of the object.

onmouseover Fires when the user moves the mouse pointer into the object.

onmouseup Fires when the user releases a mouse button while the mouse is over the object.

onmousewheel Fires when the wheel button is rotated.

onmove Fires when the object moves.

onmoveend Fires when the object stops moving.

onmovestart Fires when the object starts to move.

onpaste Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.

onpropertychange Fires when a property changes on the object.

onreadystatechange Fires when the state of the object has changed.

onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.

onresizestart Fires when the user begins to change the dimensions of the object in a control selection.

onselectionchange Fires when the selection state of a document changes.

onstop Fires when the user clicks the Stop button or leaves the Web page.

body-Events

onactivate Fires when the object is set as the active element.

onafterprint Fires on the object immediately after its associated document prints or previews for printing.

onbeforeactivate Fires immediately before the object is set as the active element.

onbeforecut Fires on the source object before the selection is deleted from the document.

onbeforedeactivate Fires immediately before the activeElement is changed from the current object to another object in the parent document.

onbeforeeditfocus Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.



onbeforepaste Fires on the target object before the selection is pasted from the system clipboard to the document.

onbeforeprint Fires on the object before its associated document prints or previews for printing.

onbeforeunload Fires prior to a page being unloaded.

onclick Fires when the user clicks the left mouse button on the object.

oncontextmenu Fires when the user clicks the right mouse button in the client area, opening the context menu.

oncontrolselect Fires when the user is about to make a control selection of the object.

oncut Fires on the source element when the object or selection is removed from the document and added to the system clipboard.

ondblclick Fires when the user double-clicks the object.

ondeactivate Fires when the activeElement is changed from the current object to another object in the parent document.

ondrag Fires on the source object continuously during a drag operation.

ondragend Fires on the source object when the user releases the mouse at the close of a drag operation.

ondragenter Fires on the target element when the user drags the object to a valid drop target.

ondragleave Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.

ondragover Fires on the target element continuously while the user drags the object over a valid drop target.

ondragstart Fires on the source object when the user starts to drag a text selection or selected object.

ondrop Fires on the target object when the mouse button is released during a drag-and-drop operation.

onfilterchange Fires when a visual filter changes state or completes a transition.

onfocusin Fires for an element just prior to setting focus on that element.

onfocusout Fires for the current element with focus immediately after moving focus to another element.

onkeydown Fires when the user presses a key.

onkeypress Fires when the user presses an alphanumeric key.

onkeyup Fires when the user releases a key.

onload Fires immediately after the browser loads the object.

onlosecapture Fires when the object loses the mouse capture.

onmousedown Fires when the user clicks the object with either mouse button.

onmouseenter Fires when the user moves the mouse pointer into the object.

onmouseleave Fires when the user moves the mouse pointer outside the boundaries of the object.

onmousemove Fires when the user moves the mouse over the object.

onmouseout Fires when the user moves the mouse pointer outside the boundaries of the object.

onmouseover Fires when the user moves the mouse pointer into the object.

onmouseup Fires when the user releases a mouse button while the mouse is over the object.

onmousewheel Fires when the wheel button is rotated.

onmove Fires when the object moves.

onmoveend Fires when the object stops moving.

onmovestart Fires when the object starts to move.

onpaste Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.

onpropertychange Fires when a property changes on the object.

onreadystatechange Fires when the state of the object has changed.

onresizeend Fires when the user finishes changing the dimensions of the object in a control selection.

onresizestart Fires when the user begins to change the dimensions of the object in a control selection.

onscroll Fires when the user repositions the scroll box in the scroll bar on the object.

onselect Fires when the current selection changes.

onselectstart Fires when the object is being selected.

onunload Fires immediately before the object is unloaded.

onabort

Fires when the user aborts the download of an image.

Bubbles No

Cancels Yes

nur Objekt IMG

onactivate

Fires when the object is set as the active element.

Bubbles Yes

Cancels No

Default action Change activation from the event.fromElement to the event.srcElement.

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onafterprint

Fires on the object immediately after its associated document prints or previews for printing.

Bubbles No

Cancels No

window, BODY, FRAMESET

onafterupdate

Fires on a databound object after successfully updating the associated data in the data source object.
 Bubbles Yes
 Cancels No
 A, BDO, BUTTON, CUSTOM, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT type=radio, INPUT type=text, LABEL, LEGEND, MARQUEE, RT, RUBY, SELECT, SPAN, TEXTAREA

onbeforeactivate
 Fires immediately before the object is set as the active element.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforecopy
 Fires on the source object before the selection is copied to the system clipboard.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, FIELDSET, FORM, hn, I, IMG, LABEL, LEGEND, LI, LISTING, MENU, NOBR, OL, P, PLAINTEXT, PRE, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TD, TEXTAREA, TH, TR, TT, U, UL

onbeforecut
 Fires on the source object before the selection is deleted from the document.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforedeactivate
 Fires immediately before the activeElement is changed from the current object to another object in the parent document.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onbeforeeditfocus
 Fires before an object contained in an editable element enters a UI-activated state or when an editable container object is control selected.
 Bubbles Yes
 Cancels Yes
 DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

onbeforepaste
 Fires on the target object before the selection is pasted from the system clipboard to the document.
 Bubbles Yes



Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onbeforeprint
 Fires on the object before its associated document prints or previews for printing.
 Bubbles No
 Cancels No
 window, BODY, FRAMESET

onbeforeunload
 Fires prior to a page being unloaded.
 Bubbles No
 Cancels Yes
 BODY, FRAMESET, window

onbeforeupdate
 Fires on a databound object before updating the associated data in the data source object.
 Bubbles Yes
 Cancels Yes
 A, BUTTON, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT type=radio, INPUT type=text, TEXTAREA, LABEL, LEGEND, MARQUEE, SELECT, SPAN, BDO, CUSTOM, RT, RUBY

onblur
 Fires when the object loses the input focus.
 Bubbles No
 Cancels No
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onbounce
 Fires when the behavior property of the marquee object is set to "alternate" and the contents of the marquee reach one side of the window.
 Bubbles No
 Cancels Yes
 MARQUEE

oncellchange
 Fires when data changes in the data provider.
 Bubbles Yes
 Cancels No
 APPLET, BDO, OBJECT

onchange
 Fires when the contents of the object or selection have changed.
 Bubbles No
 Cancels Yes
 INPUT type=text, SELECT, TEXTAREA

onclick
 Fires when the user clicks the left mouse button on the object.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

oncontextmenu
 Fires when the user clicks the right mouse button in the client area, opening the context menu.
 Bubbles Yes



Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

oncontrolselect
 Fires when the user is about to make a control selection of the object.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

oncopy
 Fires on the source element when the user copies the object or selection, adding it to the system clipboard.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, FIELDSET, FORM, hn, HR, I, IMG, LEGEND, LI, LISTING, MENU, NOBR, OL, P, PLAINTEXT, PRE, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TD, TH, TR, TT, U, UL

oncut
 Fires on the source element when the object or selection is removed from the document and added to the system clipboard.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondataavailable
 Fires periodically as data arrives from data source objects that asynchronously transmit their data.
 Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

ondatasetchanged
 Fires when the data set exposed by a data source object changes.
 Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

ondatasetcomplete
 Fires to indicate that all data is available from the data source object.
 Bubbles Yes
 Cancels No
 APPLET, OBJECT, XML

ondblclick
 Fires when the user double-clicks the object.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT



type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondeactivate
Fires when the activeElement is changed from the current object to another object in the parent document.
Bubbles Yes
Cancels No
A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

ondrag
Fires on the source object continuously during a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragend
Fires on the source object when the user releases the mouse at the close of a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragenter
Fires on the target element when the user drags the object to a valid drop target.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondragleave
Fires on the target object when the user moves the mouse out of a valid drop target during a drag operation.
Bubbles Yes
Cancels Yes
A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragover
Fires on the target element continuously while the user drags the object over a valid drop target.
Bubbles Yes
Cancels Yes



A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT
 type=button,
 INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset,
 INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P,
 PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD,
 TEXTAREA, TR, TT, U, UL, VAR, XMP

ondragstart
 Fires on the source object when the user starts to drag a text selection or selected object.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, CAPTION, CENTER, CITE, CODE, CUSTOM,
 DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT
 type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT
 type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P,
 PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD,
 TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

ondrop
 Fires on the target object when the mouse button is released during a drag-and-drop operation.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,
 CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT
 type=button,
 INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset,
 INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, P,
 PLAINTEXT, PRE, Q, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD,
 TEXTAREA,
 TR, TT, U, UL, VAR, XMP

onerror
 Fires when an error occurs during object loading.
 Bubbles No
 Cancels Yes
 IMG, OBJECT, STYLE, window

onerrorupdate
 Fires on a databound object when an error occurs while updating the associated data in the data source object.
 Bubbles Yes
 Cancels No
 A, BUTTON, DIV, FRAME, IFRAME, IMG, INPUT type=checkbox, INPUT type=hidden, INPUT type=password, INPUT
 type=radio, INPUT type=text, TEXTAREA, LABEL, LEGEND, MARQUEE, SELECT, SPAN, BDO, CUSTOM, RT, RUBY

onfilterchange
 Fires when a visual filter changes state or completes a transition.
 Bubbles No
 Cancels No
 BDO, BODY, BUTTON, CUSTOM, DIV, FIELDSET, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file,
 INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text,
 MARQUEE, nextID, RT, RUBY, SPAN, TABLE, TD, TEXTAREA, TH, TR

onfinish
 Fires when marquee looping is complete.
 Bubbles No
 Cancels Yes
 MARQUEE

onfocus
 Fires when the object receives focus.
 Bubbles No
 Cancels No
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE,
 CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR, I,
 IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,
 INPUT
 type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL,
 LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,
 SMALL,
 SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,
 window, XMP

onfocusin



- Fires for an element just prior to setting focus on that element.
 Bubbles Yes
 Cancels No
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
- onfocusout
 Fires for the current element with focus immediately after moving focus to another element.
 Bubbles Yes
 Cancels No
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
- onhelp
 Fires when the user presses the F1 key while the browser is the active window.
 Bubbles Yes
 Cancels Yes
 A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP
- onkeydown
 Fires when the user presses a key.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
- onkeypress
 Fires when the user presses an alphanumeric key.
 Bubbles Yes
 Cancels Yes
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
- onkeyup
 Fires when the user releases a key.
 Bubbles Yes
 Cancels No
 A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DEL, DFN, DIR, DIV, document, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
- onlayoutcomplete
 Fires when the print or print preview layout process finishes filling the current LayoutRect object with content from the source document.
 Bubbles Yes



Cancels Yes

BASE, BASEFONT, BGSOUND, BR, COL, DD, DIV, DL, DT, FONT, HEAD, HR, HTML, HTML Comment, LAYOUTRECT, LI, META, OL, OPTION, P, TITLE, UL

onload

Fires immediately after the browser loads the object.

Bubbles No

Cancels No

APPLET, BODY, EMBED, FRAME, FRAMESET, IFRAME, IMG, LINK, SCRIPT, window

onlosecapture

Fires when the object loses the mouse capture.

Bubbles No

Cancels No

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmousedown

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE,

TBODY,

TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseenter

Fires when the user moves the mouse pointer into the object.

Bubbles No

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, HTML, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

XMP

onmouseleave

Fires when the user moves the mouse pointer outside the boundaries of the object.

Bubbles No

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, HTML, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

XMP

onmousemove

Fires when the user moves the mouse over the object.

Bubbles Yes

Cancels No

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseout

Fires when the user moves the mouse pointer outside the boundaries of the object.

Bubbles Yes

Cancels No

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT



type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseover

Fires when the user moves the mouse pointer into the object.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmouseup

Fires when the user releases a mouse button while the mouse is over the object.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmousewheel

Fires when the wheel button is rotated.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,

CUSTOM,

DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=image, INPUT type=reset, INPUT type=password, INPUT type=radio, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onmove

Fires when the object moves.

Bubbles Yes

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onmoveend

Fires when the object stops moving.

Bubbles Yes

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR,

window,

XMP



onmovestart

Fires when the object starts to move.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onpaste

Fires on the target object when the user pastes data, transferring the data from the system clipboard to the document.

Bubbles Yes

Cancels Yes

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OL, P, PLAINTEXT, PRE, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onpropertychange

Fires when a property changes on the object.

Bubbles No

Cancels No

A, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE, COMMENT, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LEGEND, LI, LISTING, MAP, MARQUEE, MENU, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP

onreadystatechange

Fires when the state of the object has changed.

Bubbles No

Cancels No

A, ACRONYM, ADDRESS, APPLET, AREA, B, BASE, BASEFONT, BDO, BGSOUND, BIG, BLOCKQUOTE, BODY, BR, BUTTON, CAPTION, CENTER, CITE, CODE, COL, COLGROUP, COMMENT, CUSTOM, DD, DEL, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, HEAD, hn, HR, HTML, I, IFRAME, IMG, INPUT type=button,

INPUT

type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LINK, LISTING, MAP, MARQUEE, MENU, namespace, nextID, NOBR, NOFRAMES, NOSCRIPT, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SCRIPT, SELECT, SMALL, SPAN, STRIKE, STRONG, STYLE, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TITLE, TR, TT, U, UL, VAR, XML, XMP

onreset

Fires when the user resets a form.

Bubbles No

Cancels Yes

FORM

onresize

Fires when the size of the object is about to change.

Bubbles No

Cancels No

A, ADDRESS, APPLET, B, BIG, BLOCKQUOTE, BUTTON, CENTER, CITE, CODE, CUSTOM, DD, DFN, DIR, DIV, DL,

DT,

EM, EMBED, FIELDSET, FORM, FRAME, hn, HR, I, IMG, INPUT type=button, INPUT type=file, INPUT type=image,

INPUT

type=password, INPUT type=reset, INPUT type=submit, INPUT type=text, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PRE, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TEXTAREA, TT, U, UL, VAR, window, XMP

onresizeend

Fires when the user finishes changing the dimensions of the object in a control selection.

Bubbles Yes

Cancels No



A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

INPUT IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onresizestart

Fires when the user begins to change the dimensions of the object in a control selection.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, APPLET, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CUSTOM, DD, DFN, DIR, DIV, DL, document, DT, EM, EMBED, FIELDSET, FONT, FORM, FRAME, FRAMESET, hn, HR,

I,

INPUT IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image,

INPUT

type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, ISINDEX, KBD, LABEL, LEGEND, LI, LISTING, MARQUEE, MENU, OBJECT, OL, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT,

SMALL,

SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, window, XMP

onrowenter

Fires to indicate that the current row has changed in the data source and new data values are available on the object.

Bubbles Yes

Cancels No

APPLET, OBJECT, XML

onrowexit

Fires just before the data source control changes the current row in the object.

Bubbles No

Cancels Yes

APPLET, OBJECT, XML

onrowsdelete

Fires when rows are about to be deleted from the recordset.

Bubbles Yes

Cancels No

APPLET, OBJECT, XML

onrowsinserted

Fires just after new rows are inserted in the current recordset.

Bubbles Yes

Cancels No

APPLET, OBJECT, XML

onscroll

Fires when the user repositions the scroll box in the scroll bar on the object.

Bubbles No

Cancels No

APPLET, BDO, BODY, CUSTOM, DIV, EMBED, MAP, MARQUEE, OBJECT, TABLE, TEXTAREA, window

onselect

Fires when the current selection changes.

Bubbles No

Cancels Yes

BODY, INPUT type=text, TEXTAREA

onselectionchange

Fires when the selection state of a document changes.

Bubbles No

Cancels No

document

onselectstart

Fires when the object is being selected.

Bubbles Yes

Cancels Yes

A, ACRONYM, ADDRESS, AREA, B, BDO, BIG, BLOCKQUOTE, BODY, BUTTON, CAPTION, CENTER, CITE, CODE,



	CUSTOM, DD, DEL, DFN, DIR, DIV, DL, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, KBD, LABEL, LI, LISTING, MAP, MARQUEE, MENU, nextID, NOBR, OBJECT, OL, OPTION, P, PLAINTEXT, PRE, Q, RT, RUBY, S, SAMP, SELECT, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP
onstart	<p>Fires at the beginning of every loop of the marquee object.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>MARQUEE</p>
onstop	<p>Fires when the user clicks the Stop button or leaves the Web page.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>document</p>
onsubmit	<p>Fires when a FORM is about to be submitted.</p> <p>Bubbles No</p> <p>Cancels Yes</p> <p>FORM</p>
ontimeerror	<p>Fires whenever a time-specific error occurs, usually as a result of setting a property to an invalid value.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>A, ACRONYM, ADDRESS, AREA, B, BIG, BLOCKQUOTE, BUTTON, CAPTION, CENTER, CITE, CODE, DD, DEL, DIR, DIV, DL, DT, EM, FIELDSET, FONT, FORM, hn, HR, I, IFRAME, IMG, INPUT type=button, INPUT type=checkbox, INPUT type=file, INPUT type=hidden, INPUT type=image, INPUT type=password, INPUT type=radio, INPUT type=reset, INPUT type=submit, INPUT type=text, INS, KBD, LEGEND, LI, LISTING, MARQUEE, MENU, OL, OPTION, P, PLAINTEXT, PRE, Q, S, SAMP, selection, SMALL, SPAN, STRIKE, STRONG, SUB, SUP, TABLE, TBODY, TD, TEXTAREA, TFOOT, TH, THEAD, TR, TT, U, UL, VAR, XMP</p>
onunload	<p>Fires immediately before the object is unloaded.</p> <p>Bubbles No</p> <p>Cancels No</p> <p>BODY, FRAMESET, window</p>

8.3. Einzelbeschreibungen der Events (teilweise mit Beispielen)

onabort	Abbruch des Download vom Image
onactivate	<p>erzeugt wenn Objekt als aktives gesetzt wird (event.fromElement to the event.srcElement)</p> <p>aktiv setzen bedingt nicht den Erhalt des Focus:</p> <p>aber Aktivierung per Fokus-Methode möglich</p> <p>wird immer direkt vor onload erzeugt</p> <p>ab IE 5.5</p>
Beispiel	<pre> <HTML> <HEAD> <SCRIPT> function EventOnActivate() {ID_Div.innerHTML += "onactivate erkannt";} function EventOnLoad() {ID_Div.innerHTML += "onload erkannt";} </SCRIPT> </HEAD> <BODY onactivate="EventOnActivate();" onload="EventOnLoad();"> <DIV ID="ID_Div"></DIV> </BODY> </HTML> </pre>
onafterprint	erzeugt mit Ende des Druck bzw. Druckvorschau
onafterupdate	erzeugt wenn Daten in einem Datasource-Objekt erfolgreich geupdatet wurden
onbeforeactivate	erzeugt direkt vor der Aktivierung eines Objektes



onbeforecopy erzeugt direkt vor dem Kopieren einer Selektion im Objekt in das Clipboard

Beispiel

```
<HEAD>
<SCRIPT>
    var Daten= "Das sind Text-Daten";

    function Quelle_Beforecopy()
    {event.returnValue = false; }

    function Quelle_Copy()
    {window.clipboardData.setData("Text", Daten); }

    function Ziel_BeforePaste()
    {event.returnValue = false; }

    function Ziel_Paste()
    {
        ID_Input.value = window.clipboardData.getData("Text");
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecopy="Quelle_Beforecopy()"
        oncopy="Quelle_Copy()"
    >
        diesen Text kopieren
    </SPAN>
    <INPUT ID="ID_Input" onbeforepaste="Ziel_BeforePaste()"
        onpaste="Ziel_Paste()"
    >
</BODY>
```

onbeforecut erzeugt direkt vor dem Ausschneiden einer Selektion im Objekt

Beispiel 1

```
<HEAD>
<SCRIPT>
    var Kette = "";

    function EventBeforeCut()
    {event.returnValue = false; }

    function EventCut()
    {
        Kette= ID_Span.innerText;
        ID_Span.innerText = "";
        event.returnValue = false;
    }

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Div.innerText = Kette;
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        diesen Text selektieren und ausschneiden
    </SPAN>
    <BR>
    <DIV ID="ID_Div" onbeforepaste="EventBeforePaste()" onpaste="EventPaste()">
        hierher den ausgeschnittenen Text einfügen
    </DIV>
</BODY>
```

Beispiel 2

```
<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";
```




```

function Init()
{
    var TextBereich = document.body.createTextRange(); // Bereich aller Texte
    TextBereich.findText(ID_Span1.innerText); // aber den Text im ID_Span1 suchen
    TextBereich.select(); // und dann selektieren
}

function EventBeforeCut()
{
    Kette = ID_Span1.innerText;
    event.returnValue = false;
}

function EventCut()
{window.clipboardData.setData("Text", TextDaten); } // setData liefert return-Wert
// also
event.returnValue = ...; // noch nötig zu
kodieren

function EventBeforePaste()
{event.returnValue = false; }

function EventPaste()
{
    ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
    event.returnValue = false;
}
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID=" ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfuegen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

onbeforedeactivate	erzeugt direkt bevor ein Objekt als deaktiv gesetzt wird
onbeforeeditfocus	erzeugt direkt vor der User-Interaktivität auf ein editierbares Objekt immer erzeugt wenn INPUT-Objekt oder TEXTAREA-Objekt den focus erhält
onbeforepaste	erzeugt direkt vor dem Einfügen aus dem Clipboard in ein Objekt
onbeforeprint	erzeugt direkt vor dem Druck bzw. Druckvorschau
onbeforeunload	erzeugt direkt vor dem Unload eines Dokumentes unload = verlassen des Dokumentes durch Laden eines neuen Dokumentes in das selbe Fenster

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    function DokumentSchliessen()
    {event.returnValue = "Irgendeinen Stringwert liefern. Vor dem Schliessen wird Dialogbox angezeigt."}
</SCRIPT>
</HEAD>
<BODY onbeforeunload="DokumentSchliessen ()">
    <A HREF="http://www.test.de">zu www.test.de</A>
</BODY>
</HTML>

```

onbeforeupdate	erzeugt mit dem Beginn des Update von Daten eines Datasource-Objektes
onbegin	erzeugt wenn ein Element auf der Timeline aktiviert und damit die Timeline aktiv wird nicht erzeugt für eine Wiederholung der Animation des Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Element erzeugt für eine Wiederholung der Animation des Eltern-Elementes, also wenn Eigenschaft .repeatCount > 1 ist für das Eltern-Element siehe onend und onrepeat siehe Objekt currTimeState und Behavior .style.time2



Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
      BEGIN="indefinite"
      DUR="5"
      REPEATCOUNT="5"
      onbegin="alert('Start der Animation');">
  <t:ANIMATEMOTION ID="ID_Animatemotion"
    TARGETELEMENT="ID_Div"
    TO="200,0"
    BEGIN="0"
    DUR="2"
    AUTOREVERSE="true">
  </t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
      CLASS="time_line_klasse"
      STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
            border:solid black 1px;
            ">
  sich bewogender DIV
</DIV>
<BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
<BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
</BODY>
</HTML>
```

onblur erzeugt wenn Objekt den Focus verliert

Beispiel

```
<HTML>
<BODY>
<INPUT TYPE=text NAME="Test" VALUE="onblur Test" onblur="alert(event.srcElement.name)">
</BODY>
</HTML>
```

onbounce erzeugt wenn Behavior-Eigenschaft des Marquee-Objektes auf "alternate" gesetzt ist
und der Marquee-Text eine der beiden Seiten des Elternfensters erreicht hat

Beispiel 1:

```
<BODY>
<MARQUEE BEHAVIOR="alternate" WIDTH=200 LOOP=3
      onbounce="alert('onbounce-Ereignis erkannt')">
  Marquee Text
</MARQUEE>
</BODY>
```

Beispiel 2:

```
<HTML>
<BODY bgcolor="0000ff"
      onBlur="document.bgColor='ff0000'"
      onFocus="document.bgColor='0000ff'">
  Dieses Fenster ändert die Hintergrundfarbe, wenn es nicht mehr aktiv ist.
</BODY>
</HTML>
```

oncellchange erzeugt wenn Daten verändert wurden in der Datenquelle z.B. Objekt

onchange erzeugt, wenn Kontext des Objektes sich ändert

Beispiel:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
```



```

function aenderung(wert)
{
    alert("Die Eingabe hat sich geändert!\n" + "Der neue Wert ist " + wert);
}

// -->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    <INPUT TYPE="text" onchange="aenderung(this.value)">
</FORM>
</BODY>
</HTML>

```

onclick erzeugt mit Druck der linken Maustaste auf ein Objekt benötigt vorher die Eventfolge erzeugt onmousedown und onmouseup wobei die Maus auch dabei über dem Objekt sein muss bei Radio-Buttons-Gruppe: onclick event immer nach onbeforeupdate and onafterupdate events for the control group. bei focusfähigem Objekt: onfocus event erzeugt vor dem onclick event bei Doppelklick per linker Maustaste: zuerst onclick ausgelöst, dann ondblclick

oncontentready erzeugt, wenn auf die HTC-Komponente zugegriffen wird und dieses komplett geparkt ist und das Event window.onload erzeugt wurde Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden. siehe Objekt element und HTC-Datei

Beispiel:

```

<PUBLIC:ATTACH EVENT="oncontentready" ONEVENT="Anzeige()" />
<SCRIPT LANGUAGE="JScript">
    function Anzeige()
    {window.alert ('Objekt element mit innerHTML = ' + element.innerHTML); }
</SCRIPT>

```

oncontentsave erzeugt, wenn die Umgebung der HTC-Komponente gespeichert oder kopiert wird Hinweis: Änderung einer Eigenschaft der HTC-Komponente wird per Event onpropertychange angezeigt siehe Objekt element und HTC-Datei

oncontextmenu erzeugt wenn rechte Maustaste gedrückt wurde Unterdrückung, wenn der Eventhandler event.returnValue=false; enthält (mit return false; erfolgt keine Unterdrückung)

Beispiel

```

<SPAN STYLE="width:300; background-color:blue; color:white;" oncontextmenu="return false">
    <P>Das Kontextmenue ist innerhalb dieses SPAN abgeschaltet</P>
</SPAN>

```

oncontrolselect erzeugt wenn User eine Control-Selektion im Objekt tätigt (z.B. CTRL+ Maus)

oncopy erzeugt wenn Quellelement oder Selektion dem Clipboard hinzugefügt wird per rechte Maus-Click und Anwahl des Menüpunktes Copy Hinweis: Festlegung der Datenart per Methode .setData() oder CTRL-C

oncut erzeugt durch Quell-Objekt wenn Daten aus Quellobjekt in das Clipboard verschoben werden

Beispiel 1

```

<HEAD>
<SCRIPT>
    var Kette = "";

    function EventBeforeCut()
    {event.returnValue = false; }

    function EventCut()
    {
        Kette= ID_Span.innerText;
        ID_Span.innerText = "";
        event.returnValue = false;
    }

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()

```



```

    {
        ID_Div.innerText = Kette;
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        diesen Text selektieren und ausschneiden
    </SPAN>
    <BR>
    <DIV ID="ID_Div" onbeforepaste="EventBeforePaste()" onpaste="EventPaste()">
        hierher den ausgeschnittenen Text einfügen
    </DIV>
</BODY>

```

Beispiel 2

```

<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";

    function Init()
    {
        var TextBereich = document.body.createTextRange(); // Bereich aller Texte
        TextBereich.findText(ID_Span1.innerText);           // aber den Text im ID_Span1 suchen
        TextBereich.select();                               // und dann selektieren
    }

    function EventBeforeCut()
    {
        Kette = ID_Span1.innerText;
        event.returnValue = false;
    }

    function EventCut()
    { window.clipboardData.setData("Text", TextDaten); } // setData liefert return-Wert
                                                    // also
    event.returnValue = ...;                          //
                                                    //
    kodieren                                           //      noch nötig zu

    function EventBeforePaste()
    { event.returnValue = false; }

    function EventPaste()
    {
        ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID="ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfügen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

ondataavailable	erzeugt von der Datenquelle bei jedem Senden von Daten (asynchrones Senden von der Datenquelle (Objekt))
ondatasetchanged	erzeugt von der Datenquelle wenn Daten geändert wurden in der Datenquelle verwendet bei Filter-Operationen
ondatasetcomplete	erzeugt von Datenquelle wenn alle Daten der Quelle verfügbar also sendbar sind
ondblclick	erzeugt wenn Doppelklick mit Maus mit folgender Eventfolge onmousedown, onmouseup, onclick, onmouseup, and then ondblclick.
ondeactivate	erzeugt mit Deaktivierung eines Objektes



ondetach erzeugt, wenn die Event-Überwachung in einer HTC-Komponente abgeschaltet wird
siehe Objekt element und HTC-Datei

Beispiel:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT="EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
function CursorNeu()
{
    if (event.srcElement == element)
    {
        normalColor      = style.color;
        runtimeStyle.color = "red";
        runtimeStyle.cursor = "hand";
    }
}

function CursorNormal()
{
    if (event.srcElement == element)
    {
        runtimeStyle.color = normalColor;
        runtimeStyle.cursor = "";
    }
}

function EventEntfernen()
{
    detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
    detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
}

attachEvent ('onmouseover', CursorNeu);
attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

onreadystatechange erzeugt, wenn das HTML-Dokument, das die HTC-Komponente implementiert hat, komplett geparkt ist und
das Event window.onload erzeugt wurde
Parsen des HTML-Dokumentes schliesst Implementierung der HTC-Datei und Parsen der
HTC-Komponenten ein
Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden.
siehe Objekt element und HTC-Datei

ondrag erzeugt kontinuierlich während allen Drag-Operationen, aber erst nach ondragstart Event
Drag = Maus gedrückt halten

ondragend erzeugt am Ende der Dragoperation also wenn Maus losgelassen wird
erzeugt immer vom Quellobjekt
Hinweis: Zielobjekt erzeugt anschliessend ondragleave Event

ondragenter erzeugt wenn Maus über Objekt gedrückt wurde UND der User das Quell-Objekt bei gedrückter
Maustaste zieht
erzeugt immer vom Ziel-Objekt wenn Ziel erreicht wurde
ist das ERSTE Event des Ziels, also das als Erstes ausgelöste Event
erzeugt immer vom Quell-Objekt solange Ziel nicht erreicht wurde
wird immer direkt vor dem ondragover Event erzeugt

ondragleave erzeugt vom Ziel-Objekt wenn User die Maus aus dem Ziel bewegt während Drag-Operation

ondragover erzeugt vom Ziel-Objekt wenn User die Maus über das Ziel bewegt während Drag-Operation
UND immer direkt nach dem ondragenter Event

ondragstart erzeugt vom Quell-Objekt wenn User die Selektion für Drag-Operationen ausführt
auch bei Textelement
ist ERSTES Event für Drag-und Drop
Hinweis: Quell-Objekt nutzt Methode .setData() für Übergabe der Daten

ondrop erzeugt vom Ziel-Objekt wenn Maustaste losgelassen wird über dem Ziel
Drop = Ablegen
direkt erzeugt vor ondragleave and ondragend Events.

onend erzeugt wenn das aktive Element das Ende der aktiven Timeline erreicht inklusive
aller Wiederholungen laut .repeatCount
bzw. wenn das aktive Element gestoppt wird



erzeugt wenn das Elternelement das Ende seiner Timeline erreicht hat
 und somit des Kindelement ebenfalls endet
 nicht erzeugt wenn .fill mit Wert "freeze" oder "hold" für das Element kodiert wurde,
 es sei denn, das Elternelement hat das Ende seiner Timeline erreicht
 siehe Methode .endElement() und Eigenschaft .end
 siehe onbegin und onrepeat
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
BEGIN="indefinite"
DUR="5"
REPEATCOUNT="5"
onend="alert('Ende der Animation');">
<t:ANIMATEMOTION ID="ID_Animatemotion"
TARGETELEMENT="ID_Div"
TO="200,0"
BEGIN="0"
DUR="2"
AUTOREVERSE="true">
</t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
CLASS="time_line_klasse"
STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
border:solid black 1px;
">
</DIV>
<BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
<BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
</BODY>
</HTML>
```

onerror erzeugt wenn Laufzeit-Fehler beim Laden eines Objektes

Beispiel: Fehlerverfolgung

```
<SCRIPT>
function FehlerAufspuren (MeldungString, UrlString, ZielenNummerString)
{
ID_Div.innerHTML += "<B>Fehler erkannt</B>";
ID_Div.innerHTML += "Error: " + MeldungString + "<BR>";
ID_Div.innerHTML += "Line: " + ZielenNummerString + "<BR>";
ID_Div.innerHTML += "URL: " + UrlString + "<BR>";

return false;
}

window.onerror=FehlerAufspuren; // ohne () kodieren
</SCRIPT>
<DIV ID="ID_Div"></DIV>
```

Beispiel für Bild

```
<SCRIPT>
var Kette = '<IMG STYLE="display: none;" ID="ID_IMG" ALT="Das ist ein Bild ">';

function Laden()
{
ID_Div.innerHTML=Kette; // wird sofort geparkt, also IMG-Tag ausgeführt
// ( anstelle von eval()
// bzw. document.write() )

ID_IMG.src="";
ID_IMG.style.display="block";

ID_IMG.onerror= IMGAltTextAufFehlerMeldung; // ohne ()
```



```

    }

    function IMGAltTextAufFehlerMeldung ()
    {
        ID_IMG.alt="Das Bild konnte nicht geladen werden.";
        return true;
    }
</SCRIPT>
<INPUT TYPE=button VALUE="Klicke zum Bildladen" onclick="Laden()">
<DIV ID="ID_Div"></DIV>

```

onerrorupdate erzeugt wenn Fehler während Datenupdate für ein Datasource-Objekt auftrat

onfilterchange erzeugt wenn ein visueller Filter komplett abgelaufen ist

onfinish erzeugt wenn alle Durchläufe des Marquee-Objektes komplett beendet sind
Anzahl der Durchläufe laut LOOP-Attribut
LOOP-Attribut muss Wert > 1 haben, wenn onfinish erzeugt werden soll

Beispiel

```

<BODY>
<MARQUEE LOOP=2 onfinish="alert(event.srcElement.id + ' onfinish-Ereignis erkannt')">
    Marquee Text
</MARQUEE>
</BODY>

```

onfocus erzeugt, wenn Objekt den Focus erhält
Fokus nur erhaltbar nach dem kompletten Laden des Dokumentes

Beispiel für Label

```

<STYLE>
.normal {background-color:"#FFFFFF"; color:"#000000"; font-weight:normal; font-size:8pt; font-family:Arial;}
.accessible { background-color:"beige"; font-weight:bold; font-size:10pt;}
</STYLE>
<SCRIPT>
function StyleWechsel()
{
    // Input-Objekt
    event.srcElement.className="accessible"; // Input-Objekt

    // Label-Objekt
    var ZeigerAuf Label =eval(event.srcElement.id + "_Label ");
                                // ID ist "ID_Input"
                                //      also "ID_Input" + " _Label" = "ID_Input_Label"
                                //      Zeichenkettenoperation leider nötig, wenn mehrere
                                //      Objekte mit Eventerzeugung vorhanden wären
                                //      und ebenfalls nötig wegen Bezug im Label auf das
                                //      ID des Objektes, das Label haben soll
    ZeigerAuf Label.className="accessible";
}
</SCRIPT>
<LABEL FOR="ID_Input" oInput" CLASS="normal" ID="ID_Input_Label">Text eingeben</LABEL>
<INPUT TYPE="text" CLASS="normal" onfocus="StyleWechsel()" ID="ID_Input">

```

Beispiel für INPUT:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function setzeFokus()
    {
        document.meinFormular.eingabe.focus();
        document.meinFormular.eingabe.select();
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
<FORM NAME="meinFormular">
    <INPUT TYPE="text"
        NAME="eingabe"
        VALUE="Eingabe..."
    >
    <INPUT TYPE="button"

```



```
VALUE="Bitte drücken"
onclick="setzeFokus();">
```

```
</FORM>
</BODY>
</HTML>
```

onfocusin erzeugt bevor das Element den Focus erhält

onfocusout erzeugt nachdem das Element den Focus verloren hat

onhelp erzeugt wenn F1-Taste gedrückt im aktuellen Fenster

onhide erzeugt wenn der Media Bar Player gerade unsichtbar gemacht wird (versteckt wird)
siehe Eigenschaft `.enabled`
entspricht dem Schliessen des Media Bar Player durch den User
Media Bar Player ist der Windows Media Player
siehe Behavior `.style.mediaBar`

onkeydown erzeugt wenn irgend eine Tastatur-Taste gedrückt wurde

Beispiel

```
<SCRIPT>
function TastenCodeHolen()
{
    if(ID_Input.checked)
    {
        ID_Textarea.innerText+="[Keycode = " + event.keyCode + "];"
        event.returnValue=false;
    }
    else
    { ID_Textarea.innerText+=String.fromCharCode(event.keyCode); }
}
</SCRIPT>
<INPUT TYPE="checkbox" ID="ID_Input">
<INPUT TYPE="text" onkeydown="TastenCodeHolen()">
<TEXTAREA ID="ID_Textarea" ROWS="10" COLS="50"></TEXTAREA>
```

onkeypress erzeugt bei Druck einer alphanumerischen Tastatur-Taste

Beispiel

```
<HEAD>
<SCRIPT>
function ShiftPruefen()
{
    if (window.event.shiftKey)
    { ID_Input.value = "Shift erkannt"; }
}
</SCRIPT>
</HEAD>
<BODY>
    drücke SHIFT mit anderer Taste
    <INPUT TYPE="text" onkeypress="ShiftPruefen()">
    <INPUT TYPE="text" ID="ID_Input">
</BODY>
```

onkeyup erzeugt wenn gedrückte Tastatur-Taste losgelassen wird

onlayoutcomplete erzeugt wenn das Druckobjekt bzw. Druckvorschau-Objekt komplett gefüllt ist
setzt Eigenschaft `.contentOverflow` auf true

onload erzeugt mit dem Laden eines Objektes

Beispiel 1

```
<BODY>
<SCRIPT FOR=window EVENT=onload LANGUAGE="JScript">
    window.status = "Seite ist geladen!";
</SCRIPT>
</BODY>
```

Beispiel 2

```
<SCRIPT>
function Meldung()
{ window.status = "Image " + event.srcElement.src + " ist geladen"; }
</SCRIPT>
<BODY>
    <IMG SRC="test.gif" onload="Meldung()">
</BODY>
```



Beispiel 3 für Ermittlung der Verweilzeit des Users auf der Webseite

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
    var start_zeit;        // muss global sein

    function start()
    {start_zeit = new Date();}

    function ende()
    {
        var ende_zeit = new Date();
        var wert=ende_zeit.getTime() - start_zeit.getTime();
        wert = wert /1000;    // in Sekunden umrechnen
        alert("Verweilzeit in Sekunden: " + Math.floor(wert).toString());
    }
//-->
</SCRIPT>
</HEAD>

<BODY ... onLoad="start();" onUnload="ende();">
</BODY>
</HTML>
```

onlosecapture erzeugt, wenn Objekt die Mausüberwachung verliert

Beispiel

```
<BODY onload="ID_Div.setCapture()"
onclick="ID_Div.releaseCapture();"
>
    <DIV ID="ID_Div"divOwnCapture
        onmousemove="ID_Textarea.value=event.clientX + event.clientY"; // kein alert() !!!!!
        onlosecapture="alert(event.srcElement.id + ' ohne Maus-Ueberwachung')";
    >
        Ueber diesen Text mit der Maus fahren
        <BR>
        <TEXTAREA ID="ID_Textarea" COLS=2></TEXTAREA>
    </DIV>
    <DIV>
        Klick hier fuer Event onlosecapture per Methode .releaseCapture()
    </DIV>
</BODY>
```

onmediacomplete erzeugt wenn das Medium zum Element (Media-Element) komplett geladen wurde für Animation per Timeline
Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer Datei (streaming media file) z.B. Gif-Bild, Video etc.
sinnvoll für Start des Elementes auf der Timeline
siehe onmediaerror
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:VIDEO ID="ID_Video"
        SRC="test.wmv"
        onmediacomplete="ID_Span.innerText +=
            'Datei test.wmv wurde komplett geladen !'
            "
    >
    </t:VIDEO>
    <BR>
    <SPAN ID="ID_Span"></SPAN>
</BODY>
</HTML>
```

onmediaerror erzeugt wenn das Medium zum Element (Media-Element) fehlerhaft geladen wurde



oder das Medium einen Fehler enthält
 für Animation per Timeline
 Media-Element benötigt vor seiner Animation dem Datenfluss des Mediums aus einer
 Datei (streaming media file) z.B. Gif-Bild, Video etc.
 sinnvoll für Start des Elementes auf der Timeline
 ersetzt das Ereignis onmedialoadfailed, das deprecated ist und nicht mehr verwendet
 werden darf !
 siehe onmediacomplete
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:IMG ID="ID_Img"
SRC="test.gif"
STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
onmediaerror="alert(' Datei test.gif nicht ladbar !')">
</t:IMG>
<t:ANIMATION ID="ID_Animation"
TARGETELEMENT="ID_Img"
TO="0,400"
DUR="2"
BEGIN="0"
ACCELERATE="1"
AUTOREVERSE="true"
REPEATCOUNT="5">
</t:ANIMATION>
</BODY>
</HTML>
```

onmousedown erzeugt wenn irgendeine Maustaste gedrückt wird
 onmouseenter erzeugt wenn Maus ein Objektbereich betritt
 onmouseleave erzeugt wenn Maus gerade den Objektbereich verlässt
 onmousemove erzeugt wenn Maus im Bereich eines Objekt bewegt wird

Beispiel

```
<SCRIPT>
function Anzeige()
{ID_Span.innerText="Coords: (" + event.clientX + ", " + event.clientY + ")";} // kein alert() !!!!
</SCRIPT>
<DIV onmousemove="Anzeige()">
<SPAN ID="ID_Span"></SPAN>
</DIV>
```

onmouseout erzeugt wenn Maus den Bereich eines Objektes gerade verlässt
 genau 1x erzeugt pro Verlassen
 onmouseover erzeugt wenn Maus den Bereich eines Objektes gerade betritt
 genau 1x erzeugt pro Betreten
 onmouseup erzeugt wenn Maustaste losgelassen wird
 Hinweis: Eigenschaft .button liefert die losgelassene Taste
 onmousewheel erzeugt wenn Mousrad gedreht wird
 mit dem Drehen wird Eigenschaft .wheelDelta gefüllt
 mit Integer-Vielfachen von 120 Grad Umdrehung
 wenn > 0 so Drehung vom User weg
 wenn < 0 so Drehung zum User hin
 ab IE 6.0

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
var ZoomFaktorStartWert = 10; // > 0, ganzzahlig, in Prozent
var ZoomSchrittWeite = 1; // > 0, ganzzahlig
```



```

var DrehSchrittWeite = 1; // 1 oder 2 oder 3 da Faktor von 120 Grad

var ZoomFaktor = ZoomFaktorStartWert; // > 0, ganzzahlig, in Prozent
function VergroessernVerkleinern()
{
    if (event.wheelDelta >= (DrehSchrittWeite * 120))
    { ZoomFaktor += ZoomSchrittWeite; }
    else
    {
        if (event.wheelDelta <= (-1 * (DrehSchrittWeite * 120)))
        { ZoomFaktor -= ZoomSchrittWeite; }
    }

    if (ZoomFaktor <= 0)
    { ZoomFaktor = ZoomFaktorStartWert; }

    ID_Image.style.zoom = ZoomFaktor + '0%';
}

</SCRIPT>
</HEAD>
<BODY>
    <IMG ID="ID_Image" SRC="test.jpg" onmousewheel="VergroessernVerkleinern()">
</BODY>
</HTML>

```

onmove

erzeugt, wenn Objekt bewegt wird:

Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute;**")

nicht erzeugt, wenn ein Unterobjekt bewegt wird:

Unterobjekt muss **eigenen** Handler haben

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
function HandleFuerOnMoveStart()
{
    var ZeigerAufObjektMitEvent = event.srcElement; // anstelle des ID vom DIV
                                                    // falls mehrere bewegbare
                                                    // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandleFuerOnMoveEnd()
{
    var ZeigerAufObjektMitEvent = event.srcElement; // anstelle des ID vom DIV
                                                    // falls mehrere bewegbare
                                                    // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

document.execCommand("2D-position",false,true); // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY onmovestart="HandleFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandleFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offsetTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
            bewegbarer DIV
        </DIV>
    </DIV>

```



```
</BODY>
</HTML>
```

onmoveend

erzeugt, wenn das Bewegen eines Objektes endet:

Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute;**")

nicht erzeugt, wenn ein Unterobjektbewegung endet:

Unterobjekt muss **eigenen** Handler haben

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
function HandleFuerOnMoveStart()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandleFuerOnMoveEnd()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

document.execCommand("2D-position",false,true);    // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY onmovestart="HandleFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandleFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offsetTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>
```

onmovestart

erzeugt, wenn das Bewegen eines Objektes beginnt:

Objekt muss **absolute positioniert** sein (im STYLE per "**position:absolute;**")

nicht erzeugt, wenn ein Unterobjektbewegung beginnt:

Unterobjekt muss **eigenen** Handler haben

Beispiel

```
<HTML>
<HEAD>
<SCRIPT>
function HandleFuerOnMoveStart()
{
    var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                        // falls mehrere bewegbare
                                                        // Objekte vorhanden sind

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
```



```

        ID_Span2.innerHTML = event.srcElement.offsetTop;
    }

    function HandleFuerOnMoveEnd()
    {
        var ZeigerAufObjektMitEvent = event.srcElement;    // anstelle des ID vom DIV
                                                           // falls mehrere bewegbare
                                                           // Objekte vorhanden sind
        ZeigerAufObjektMitEvent.style.backgroundColor = "red";
        ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
    }

    document.execCommand("2D-position",false,true);    // 2-D Positionierung einschalten
</SCRIPT>
</HEAD>
<BODY onmovestart="HandleFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandleFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>
    offserTop = <SPAN ID="ID_Span2"></SPAN>
    <BR>
    <DIV CONTENTEDITABLE="true">
        <DIV STYLE="position:absolute;width:300px;height:100px; background-color:red;">
            bewegbarer DIV
        </DIV>
    </DIV>
</BODY>
</HTML>

```

onopenstatechange erzeugt, wenn Media Bar Player den Status bezüglich Playliste, Codec, Lizenz und Individualisierung ändert
 siehe Eigenschaft .openState
 Media Bar Player ist der Windows Media Player
 siehe Behavior .style.mediaBar

Beispiel:

```

<DIV ID="ID_Div"
STYLE="behavior:url(#default#mediaBar)"
onopenstatechange="alert(ID_Div.openState);"
>
</DIV>
<INPUT TYPE=button
VALUE='abspielen von test.asx'
onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
ID_Div.disabledUI = true;
"
>

```

onoutofsync erzeugt wenn das Element seine Timeline verliert, also nicht mehr auf seiner Timeline animiert wird (nicht synchron zur Timeline des Elementes)
 sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked":
 Eventhandler sollte neu synchronisieren z.B. Reset des Elementes durch .resetElement()
 siehe onsyncstored
 siehe Objekt currTimeState und Behavior .style.time2

onpaste erzeugt vom Zielobjekt wenn Daten aus Clipboard in das Objekt eingefügt werden

Beispiel 1

```

<HEAD>
<SCRIPT>
    var Kette = "";

    function EventBeforeCut()
    { event.returnValue = false; }

    function EventCut()
    {
        Kette= ID_Span.innerHTML;
        ID_Span.innerHTML = "";
        event.returnValue = false;
    }

    function EventBeforePaste()

```



```

        {event.returnValue = false; }

        function EventPaste()
        {
            ID_Div.innerText = Kette;
            event.returnValue = false;
        }
    </SCRIPT>
</HEAD>
<BODY>
    <SPAN ID="ID_Span" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        diesen Text selektieren aund ausschneiden
    </SPAN>
    <BR>
    <DIV ID="ID_Div" onbeforepaste="EventBeforePaste()" onpaste="EventPaste()">
        hierher den ausgeschnittenen Text einfügen
    </DIV>
</BODY>

```

Beispiel 2

```

<HEAD>
<SCRIPT>
    var TextDaten = "new content associated with this object";
    var Kette = "";

    function Init()
    {
        var TextBereich = document.body.createTextRange(); // Bereich aller Texte
        TextBereich.findText(ID_Span1.innerText);           // aber den Text im ID_Span1 suchen
        TextBereich.select();                               // und dann selektieren
    }

    function EventBeforeCut()
    {
        Kette = ID_Span1.innerText;
        event.returnValue = false;
    }

    function EventCut()
    {window.clipboardData.setData("Text", TextDaten); }      // setData liefert return-Wert
                                                                // also
    event.returnValue = ...;                                  //
                                                                //
                                                                //      nocht nötig zu
                                                                //      kodieren

    function EventBeforePaste()
    {event.returnValue = false; }

    function EventPaste()
    {
        ID_Span2.value = window.clipboardData.getData("Text", TextDaten);
        event.returnValue = false;
    }
</SCRIPT>
</HEAD>
<BODY onload="Init()">
    <SPAN ID="ID_Span1" onbeforecut="EventBeforeCut()" oncut="EventCut()">
        Diesen Text ausschneiden
    </SPAN>
    <INPUT ID=" ID_Span2" TYPE="text" VALUE="Den ausgeschnittenen Text hier einfuegen"
        onbeforepaste="EventBeforePaste()"
        onpaste="EventPaste()"
    >
</BODY>

```

onpause erzeugt wenn das aktive Element auf der aktiven Timeline gerade beginnt zu pausieren
auch bei body Objekt
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }

```



```

</STYLE>
</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                      SRC="test.wmv"
                      onpause="ID_Span.innerText = "Pause !"
                      onmediacomplete="ID_Span.innerText =
                                      "Datei test.wmv wurde komplett geladen !"
    >
    </t:VIDEO>
    <BR>
    <SPAN ID="ID_Span"></SPAN>
</BODY>
</HTML>

```

onplaystatechange erzeugt, wenn Media Bar Player den Status bezüglich Wiedergabe ändert
 siehe Eigenschaft .playState
 Media Bar Player ist der Windows Media Player
 siehe Behavior .style.mediaBar

Beispiel:

```

<DIV ID="ID_Div"
      STYLE="behavior:url(#default#mediaBar)"
      onplaystatechange="alert(ID_Div.playState);"
>
</DIV>
<INPUT TYPE=button
        VALUE='abspielen von test.asx'
        onclick=" ID_Div.playURL('http://www.test.de/test.asx','video/x-ms-asf');
                  ID_Div.disabledUI = true;
        "
>

```

onpropertychange erzeugt wenn eine Objekteigenschaft geändert wird
 außer Änderung von .innerText und .innerHTML

Beispiel

```

<HEAD>
<SCRIPT>
    function ValueAendern()
    { ID_Input1.value = "Neuer Wert von VALUE";}

    function FarbeAendern()
    { ID_Input2.style.backgroundColor = "aqua";}

    function Anzeige()
    {alert(event.propertyName + " wurde verändert");}

</SCRIPT>
</HEAD>
<BODY>
    <INPUT TYPE=button ID="ID_Input1"
          VALUE="Click um VALUE zu ändern"
          onclick="ValueAendern()"
          onpropertychange="Anzeige()"
    >
    <INPUT TYPE=button ID="ID_Input2"
          VALUE="Click um Farbe zu ändern"
          onclick="FarbeAendern()"
          onpropertychange="Anzeige()"
    >
</BODY>

```

onpropertychange erzeugt, wenn die Methode **ID_HTC_Komponente.fireChange()** in der Funktion laut Attribut PUT
 aufgerufen wird (Aufruf muss programmiert werden), wobei die Funktion eine Eigenschaft der
 HTC-Komponente mit neuem Wert belegt
 siehe Objekt element und HTC-Datei

onreadystatechange erzeugt wenn Status eines Objektes sich ändert
 immer erzeugt von datenladenden folgender Objekte:
 applet, document, frame, frameSet, iframe, img, link, object, script und xml elements.

Beispiel

```

function Preufen ()
{
    if (document.readyState=="complete")

```



```

    {alert('Dokument komplett geladen und mit den Daten initialisiert.);}
}

```

document.onreadystatechange=Preufen; // ohne () kodieren

onrepeat

erzeugt mit Start **jeder** Wiederholung der Animation des aktiven Elementes
 nicht erzeugt beim Start des aktiven Elementes (siehe onbegin), also des
 ersten Durchlaufes, der keine Wiederholung ist
 nicht erzeugt durch Kind des Elementes, wenn Kind nicht selbst einen
 Handler für onrepeat kodiert hat (kein Heraufreichen des
 Ereignisses onrepeat zum Elternelement)
 .repeatCount bzw. .repeat muss > 1 sein:
 onrepeat wird also .repeatCount -1 mal erzeugt
 Kind eines Elementes
 siehe onend, onbegin, .repeatCount und .repeat
 siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:IMG ID="ID_Img"
        SRC="test.gif"
        STYLE="position:relative;top:25px;left:50px;height:100px;width:100px;"
        onmediaerror="alert('Datei test.gif nicht ladbar !')"
        onrepeat="alert('Aktuelle Wiederholung: ' + ID_Animation.currTimeState.repeatCount);"
    >
    </t:IMG>
    <t:ANIMATION ID="ID_Animation"
        TARGETELEMENT="ID_Img"
        TO="0,400"
        DUR="2"
        BEGIN="0"
        ACCELERATE="1"
        AUTOREVERSE="true"
        REPEATCOUNT="5"
    >
    </t:ANIMATION>
</BODY>
</HTML>

```

onreset

erzeugt wenn Formular zurückgesetzt wird
 Resetaktion erzeugbar per
 INPUT TYPE="reset"
 BUTTON TYPE="reset"

Beispiel

```

<HTML>
<HEAD>
<SCRIPT>
    function Anzeige()
    { ID_Span.innerHTML += "Anzeige zum Formular Reset";}
</SCRIPT>
</HEAD>
<BODY>
    <FORM NAME="Formular" ID="ID_Formular" onreset="Anzeige();"
        <INPUT TYPE="text" NAME="InputText" VALUE="">
        <BR>
        <INPUT TYPE="reset" VALUE="Formular Reset">
        <BUTTON onclick="ID_Formular.reset();">Formular Reset</BUTTON>
    </FORM>
    <SPAN ID="ID_Span"></SPAN>
    <BR>
    <BUTTON onclick="location.reload(true);">Refresh der Seite</BUTTON>
</BODY>
</HTML>

```

onreset

erzeugt wenn Element zurückgesetzt wurde per Methode .resetElement(),
 also die Timeline des Elementes den aktuellen Wert laut .begin wieder erreicht
 und damit zurückgesetzt wurde
 also nicht beim Initialisieren des Elementes und seiner Timeline
 durch Instanziierung des Elementes



siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
<t:EXCL ID="ID_Excl"
      BEGIN="indefinite"
      DUR="5"
      REPEATCOUNT="5"
      onreset="alert('Rücksetzen der Animation und Timeline');">
  <t:ANIMATEMOTION ID="ID_Animatemotion"
    TARGETELEMENT="ID_Div"
    TO="200,0"
    BEGIN="0"
    DUR="2"
    AUTOREVERSE="true"
  >
  </t:ANIMATEMOTION>
</t:EXCL>
<DIV ID="ID_Div"
      CLASS="time_line_klasse"
      STYLE="position:relative;top:15px;left:25px;height:100px;width:100px;
            border:solid black 1px;
            ">
  >
  sich bewegender DIV
</DIV>
<BUTTON onclick=" ID_Animatemotion.beginElement();">Start</BUTTON>
<BUTTON onclick=" ID_Animatemotion.endElement();">Stop</BUTTON>
<BUTTON onclick=" ID_Animatemotion.resetElement();">Reset</BUTTON>

</BODY>
</HTML>
```

onresize erzeugt, wenn Objektgröße verändert wird
nicht für embedded-Objekt

Beispiel: Reload des Dokumentes nach Resize des Browserfensters

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function neuLaden()
{
    if (document.all)
    {history.go(0);}
}
// -->
</SCRIPT>
</HEAD>
<BODY onResize=neuLaden();">
</BODY>
```

onresizeend erzeugt, wenn Änderung der Objektgröße endet

onresizestart erzeugt wenn Veränderung Objektgröße endet

onresume erzeugt wenn die Pause eines aktiven Elementes, das auf der Timeline pausiert, beendet wird
siehe Objekt currTimeState und Behavior .style.time2
auch für body Objekt

Beispiel:

```
<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
.time_line_klasse { behavior: url(#default#time2) }
</STYLE>
```



```

</HEAD>
<BODY>
    <t:VIDEO          ID="ID_Video"
                     SRC="test.wmv"
                     onpause="ID_Span.innerText = "Pause !"
                     onresume="ID_Span.innerText = "Pause beendet ""
                     onmediacomplete="ID_Span.innerText =
                                     "Datei test.wmv wurde komplett geladen !"
    >
    </t:VIDEO>
    <BR>
    <SPAN ID="ID_Span"></SPAN>
</BODY>
</HTML>

```

onreverse erzeugt wenn das aktive Element rückwärts auf der Timeline animiert wird
(auch wenn es sich um eine Wiederholung handelt, also .repeatCount > 0 ist)
.autoReverse muss auf "true" stehen
siehe Objekt currTimeState und Behavior .style.time2

Beispiel:

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
</HEAD>
<BODY>
    <t:EXCL  AUTOREVERSE="true"
            DUR="6"
            REPEATCOUNT="indefinite"
            onreverse="alert('Rückwärts auf der Timeline);"
    >
        <DIV CLASS="time_line_klasse" BEGIN="0" DUR="2">Zeile 1</DIV>
        <DIV CLASS="time_line_klasse" BEGIN="2" DUR="2">Zeile 2</DIV>
        <DIV CLASS="time_line_klasse" BEGIN="4" DUR="2">Zeile 3</DIV>
    </t:EXCL>
</BODY>
</HTML>

```

onrowenter erzeugt wenn neue Daten verfügbar sind in der aktuellen Spalte
aufgrund Änderung der Daten in der Datenquelle zu Spalte
nur für databound-Objekte, die selbst Daten halten (Quelle und Ziel identisch)

onrowexit erzeugt direkt Spaltenwechsel, also vor dem Verlassen der aktuellen Spalte
nur für databound-Objekte, die selbst Daten halten (Quelle und Ziel identisch)

onrowsdelete erzeugt direkt vor dem Löschen von Spalten

onrowsinserted erzeugt direkt nach dem Einfügen einer Spalte per Methode .AddNew()

onsave erzeugt, wenn Webseite als Datei gespeichert wird
oder Webseite als Bookmark in die Favoritenliste eingetragen wird
oder Webseite verlassen wird

Beispiel

```

<HTML>
<HEAD>
<META NAME="save" CONTENT="favorite">
<STYLE>
    .saveFavorite { behavior:url(#default#savefavorite);}
</STYLE>
</HEAD>
<BODY>
    <INPUT  TYPE=text ID="ID_Input" CLASS=saveFavorite
            onsave="javascript:alert('Event onsave erkannt');"
    >
</BODY>
</HTML>

```

onscroll erzeugt wenn User die Scrollpfeile benutzt (nicht den Scrollbalken)

onseek erzeugt wenn eine Seek-Methode zum Element aktiviert wurde:
.seekActiveTime()



```

        .seekSegmentTime()
        .seekTo()
        .seekToFrame()
    siehe Objekt currTimeState und Behavior .style.time2

```

Beispiel :

```

<HTML XMLNS:t="urn:schemas-microsoft-com:time">
<HEAD>
<?IMPORT namespace="t" implementation="#default#time2">
<STYLE>
    .time_line_klasse { behavior: url(#default#time2) }
</STYLE>
<SCRIPT>
    var FrameAnzahl=600;

    function Suchen()
    {
        // prüfen ob Element nicht aktiv ist
        if (!ID_Video.currTimeState.isActive)
        {
            // nicht aktiv, also starten
            ID_Video.beginElement();
        }
        else
        {
            // prüfen des Eingabewertes
            if( ( isFinite(ID_Input.value) )
                && (ID_Input.value < FrameAnzahl )
                && (ID_Input.value > 0)
            )
            {
                // ist okay, also ab Zeitpunkt animieren
                ID_Video.seekToFrame(ID_Input.value);
            }
            else
            {
                // fehlerhaft
                alert( "Fehler ! Frame-Wert muss > 0 und < "
                    + FrameAnzahl
                    + " sein"
                );
                ID_Input.focus();
            }
        }
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <SPAN CLASS="time_line_klasse"
        DUR=".01"
        REPEATCOUNT="indefinite"
        FILL="hold"
        onrepeat="innerText=parseInt(ID_Video.currentFrame);"
    >
        0
    </SPAN>
    <BR>
    <t:VIDEO ID="ID_Video"
        SRC="test.avi"
        STYLE="width:175px; height:150px;"
        onmediacomplete="ID_Span.innerText= ID_Video.mediaDur;"
        onseek="alert('Der Frame ' + ID_Input.value + ' wird eingestellt !')
    >
    </t:VIDEO>
    <BR>
    Dauer des AVI:
    <SPAN ID="ID_Span"></SPAN>
    &nbsp;&nbsp;&nbsp;Sekunden
    <BR>
    setze seekToFrame:
    <INPUT TYPE="text"
        VALUE=""
        NAME="ID_Input"
        SIZE="3"
    >&nbsp;&nbsp;&nbsp;Sekunden

```



```

<BR>
<BUTTON onclick="Suchen();">
    Klick fuer Seek
</BUTTON>
<BUTTON onclick=" ID_Video.beginElement()">
    Restart
</BUTTON>
</BODY>
</HTML>

```

onselect erzeugt wenn etwas selektiert wird
 Beispiel für INPUT:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    function setzeFokus()
    {
        document.meinFormular.eingabe.focus();
        document.meinFormular.eingabe.select();
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
<FORM NAME="meinFormular">
    <INPUT TYPE="text"
        NAME="eingabe"
        VALUE="Eingabe..."
    >
    <INPUT TYPE="button"
        VALUE="Bitte drücken"
        onclick="setzeFokus();">
</FORM>
</BODY>
</HTML>

```

onselectionchange erzeugt wenn Selektionsstatus im Dokument verändert wird

onselectstart erzeugt wenn Objekt selektiert wird

onshow erzeugt wenn der Media Bar Player gerade sichtbar gemacht wird (nicht versteckt wird)
 siehe Eigenschaft .enabled
 entspricht dem Öffnen des Media Bar Player durch den User
 Media Bar Player ist der Windows Media Player
 siehe Behavior .style.mediaBar

onstart erzeugt mit Beginn eines jedem Loop-Durchlaufes des des Marquee-Objektes
 Anzahl der Durchläufe laut LOOP-Attribut
 LOOP-Attribut muss Wert > 0 haben, wenn onstart erzeugt werden soll

Beispiel

```

<BODY>
<MARQUEE BEHAVIOR="alternate" LOOP=2 onstart="alert('onstart-Ereignis erkannt')">
    Marquee Text
</MARQUEE>
</BODY>

```

onstop erzeugt wenn STOP-Button im Browser-Menü gedrückt wurde
 wird direkt nach den onbeforeunload Event erzeugt
 wird vor dem onunload Event erzeugt, da die Webseite verlassen wird

Beispiel

```

var TimerID;

function Rekursion()
{
    // nach belieben etwas tun
}

function TimerLoeschen()
{window.clearInterval(TimerID); }

function Init()
{TimerID = window.setInterval("Rekursion()",1000); }

```



```
document.onstop= TimerLoeschen;      // ohne ()
window.onload=Init;                  // ohne ()
```

onsubmit erzeugt wenn das Formular gesendet wird
 Eventhandler **muss** einen Rückkehrcode liefern (true oder false) z.B. per return-Anweisung
 Sendeaktion erzeugbar per

 Art der Formularaktion laut ACTION-Attribut des Formular-Tag
 Senden unterbindbar, wenn Eventhandler false liefert als Rückkehrcode (muss programmiert werden)
 z.B. wenn Daten im Formular fehlerhaft vom User eingegeben wurden

Beispiel

```
<BODY>
  <FORM NAME="Formular" onsubmit="return(SubmitEventHandler());"></FORM>
</BODY>
```

onsyncrestored erzeugt wenn Element gerade mit seiner Timeline synchronisiert wird
 nach vorausgegangenem Abbruch der Synchronisation
 siehe onoutofsync
 sinnvoll nur bei Zwangssynchronisierung per .syncBehavior auf "locked"
 siehe Objekt currTimeState und Behavior .style.time2

ontimeerror erzeugt bei Zeitfehler nur für BODY-Objekt bei benutzung der Timeline

Beispiel

```
<SCRIPT FOR="BODY" event="ontimeerror">
  alert("HTML+TIME error erkannt");
</SCRIPT>
```

ontrackchange erzeugt wenn der Track einer Advanced Stream Redirector (ASX) Datei (*.asx)
 in der Playliste gewechselt wurde

Beispiel für Aufbau einer ASX-Datei:

```
<ASX Version="1.0" PreviewMode="No" >
  <ENTRY>
    <TITLE>First title</TITLE>
    <AUTHOR>Test</AUTHOR>
    <COPYRIGHT>2002</COPYRIGHT>
    <ABSTRACT>WAV File</ABSTRACT>
    <REF href="" > </REF>
    <banner href = "first_title.gif" >
      <moreinfo href = "first_title.doc"><moreinfo>
      <abstract>Visit the first abstract Web site</abstract>
    </banner>
  </ENTRY>
</ASX>
```

siehe Objekt currTimeState und Behavior .style.time2

onunload erzeugt mit dem direkt vor dem Unload eines Dokumentes
 unload = verlassen des Dokumentes durch Laden eines neuen Dokumentes in das selbe Fenster

Beispiel

```
<HEAD>
<SCRIPT FOR=window EVENT=onunload>
  alert("Event onunload erkannt");
</SCRIPT>

<SCRIPT>
  function Umlenken()
  {location.href="test.htm";}
</SCRIPT>
</HEAD>
<BODY>
  <INPUT TYPE=button VALUE="Klicken ... weiter zu TEST.HTM" onclick="Umlenken()">
  <IMG SRC="test.gif">
</BODY>
```

Beispiel für Ermittlung der Verweilzeit des Users auf der Webseite

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
  var start_zeit;      // muss global sein

  function start()
```



```
{start_zeit = new Date();}

function ende()
{
    var ende_zeit = new Date();
    var wert=ende_zeit.getTime() - start_zeit.getTime();
    wert = wert /1000;    // in Sekunden umrechnen
    alert("Verweilzeit in Sekunden: " + Math.floor(wert).toString());
}

//-->
</SCRIPT>
</HEAD>

<BODY ... onLoad="start();" onUnload="ende();">
</BODY>
<HTML>
```

onURLFlip

erzeugt wenn ein Scriptkommando, das in einer Advanced Streaming Format (ASF)-Datei (*.asf) liegen, ausgeführt wird
siehe Objekt currTimeState und Behavior .style.time2
Hinweis: window.event.URL Url laut einem Kommando aus einer Advanced Streaming Format (ASF)-Datei von Element auf der Timeline
es muss Ereignis onURLFlip aufgetreten sein
siehe Objekt currTimeState und Behavior .style.time2



9. Anhang: Filter des Internet Explorer

Nachfolgende Eigenschaften werden nicht von allen Filtern benötigt. Eigenschaften werden z.T. mit identischen Bezeichnern anders wertmässig belegt (je nach Filter).

.add	Überlagerung des Filters über das Originalbild
.bands	Anzahl der Streifen
.Bias	Prozentsatz der Farbwerverhöhung je höher der Bildkontrast um so geringer der sichtbare Effekt
.color	Farbe
.colorSpace	Pfad der *.icm-Datei
.direction	Filterrichtung nicht Filter Blinds und CheckerBoard

Beispiel

```
<SCRIPT>
function Setze()
{
    with (window.event.srcElement.filters[0])
    {
        if (strength < 100)
        {
            strength += 1;
            direction += 45;
        }
    }
}
</SCRIPT>
<IMG SRC="test.gif" onfilterchange="Setze()"
STYLE="filter:progid:DXImageTransform.Microsoft.MotionBlur(STRENGTH=1, DIRECTION=0)"
>
```

.Direction	Richtung nur Filter Blinds und CheckerBoard
------------	--

.duration	Dauer der Animation des Filters
-----------	---------------------------------

Beispiel

```
<SCRIPT LANGUAGE=JavaScript>
var StartBild = "/graphics/test1.jpg";
var EndeBild = "/graphics/test2.jpg";

function FilterAusfuehren()
{
    ID_Img.filters.item(0).Apply();
    ID_Img.src = EndeBild;
    EndeBild = StartBild;           // neues Endebild
    StartBild = ID_Img.src;         // altes Endebild
    ID_Img.filters.item(0).Play();
}
</SCRIPT>
<IMG ID="ID_Img" SRC="/graphics/test1.jpg";
style="width:200; height:200; filter:progid:DXImageTransform.Microsoft.Blinds(Duration=2)"
>
<BR>
<INPUT TYPE="button" value="Start Transition" onClick="FilterAusfuehren()">
```

.Dx	X-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft .SizingMethod mit Wert "auto expand"
-----	---

.Dy	Y-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft .SizingMethod mit Wert "auto expand"
-----	---

.enabled	Filter-Anwendbarkeit
----------	----------------------

Beispiel

```
<IMG ID="ID_Img"
SRC="/graphics/test.gif"
STYLE="filter:blur(STRENGTH=50) flipv()"
onmouseover="ID_Img.filters.flipv.enabled = false;"
onmouseout ="ID_Img.filters.flipv.enabled = true;"
>
```



.EndColor	Ende-Deckungskraft als Integer
.EndColorStr	Ende-Deckungskraft als String
.FilterType	Filtertyp
.finishOpacity	Deckungskraft, mit der Filter endet
.finishX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft endet
.finishY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft endet
.freq	Anzahl der Wellen
.function	Nummer der Komposition
.gradientSize	Faktor der Objektbreite als Gradientbreite
.GradientType	Orientierung des Gradienten
.grayScale	Grauskala ein/aus
Beispiel	<pre> <DIV ID="ID_Div" STYLE="position:absolute; left:270px;" > </DIV> <BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(grayScale=1)'"> Gray </BUTTON>
 <BUTTON onclick="ID_Div.style.filter=''"> Clear Filter </BUTTON> </pre>
.gridSizeX	Anzahl Spalten des Gitters
.gridSizeY	Anzahl Zeilen des Gitters
.intent	Farbverwendungsart
.Invert	Komplementärfarben ein/aus
Beispiel	<pre> <DIV ID="ID_Div" STYLE="position:absolute; left:270px;" > </DIV> <BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(invert=1)'"> Invert </BUTTON>
 <BUTTON onclick="ID_Div.style.filter=''"> Clear Filter </BUTTON>
 </pre>
.irisStyle	Schärfe des Filters
.lightStrength	prozentuale Differenz der Licht-Intensität zwischen Wellengipfel und Wellentäler
.M11	Matrixfeld M11
.M12	Matrixfeld M12
.M21	Matrixfeld M21
.M22	Matrixfeld M22
.makeShadow	Schatten ein/aus
Beispiel	<pre> <DIV STYLE="position:absolute; left:70px; filter: progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true', ShadowOpacity=1.0)" > </DIV> </pre>



.Mask Transparenzänderung auf Wert laut Eigenschaft .MaskColor ein/aus

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>
```

.MaskColor Farbmaske

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>
```

.maxSquare Maximale Anzahl der Pixels im Quadrat

.mirror Rervers ein/aus

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mirror=1)'">
    Rotate 270 degrees
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter=''">Clear Filter</BUTTON>
```

.motion Bewegungsrichtung
nicht Filter GradientWipe und Strips

.motion Art der Bewegung
nur Filter GradientWipe

.motion Ecke
nur Filter Strips

.offX horizontaler Abstand des Schatten vom Objekt UND Schattenrichtung

.offY vertikaler Abstand des Schatten vom Objekt UND Schattenrichtung

.opacity Deckungskraft mit der der Filter startet
nicht Filter BasicImage

.opacity Deckungskraft- bz.w Transparenz-Niveau (Opacity-Niveau)
nur Filter BasicImage

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(opacity=.2)'">
    Opaque
```



```

</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>
Klick Opaque-Button für Transparenz

```

.orientation Orientierung

.overlap Anteil der Überlappungs-Dauer am Wert laut Eigenschaft .duration

Beispiel

wenn .overlap auf 0.4 und .duration auf 10 Sekunden
dann 40 % der 10 Sekunden findet überlappen statt = 4 Sekunden
60 % der 10 Sekunden ohne überlappen = 6 Sekunden, also
30 % der 10 Sekunden nur Fadeout = 3 Sekunden
30 % der 10 Sekunden nur Fadein = 3 Sekunden

also Ablauf:

Sekunde 1-3	Fadeout aber nicht komplett	
Sekunde 4-7	Fadeout komplett UND Fadein beginnt	= Overlap
Sekunde 8-10	Fadein komplett	

.Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll

Beispiel

```

<SCRIPT>
function FilterSetzen()
{
    ID_Div.filters[0].Apply();

    // Achtung: nach Applay() kodierte Veränderungen vom DIV werden ERST
    // mit der Abarbeitung der Methode .play() sichtbar
    ID_Div.style.backgroundColor="blue";
    ID_Div.filters[0].percent=50;
    ID_Div.filters[0].enabled=true;
}
</SCRIPT>
<BUTTON onclick="FilterSetzen(); onclick="">FilterSetzen</BUTTON>
<BR>
<DIV ID="ID_Div"
STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.checkerboard(,duration=5, transition=7);"
>
</DIV>

```

.phase prozentuale Phasenverschiebung der Wellen,
also Verschiebung der Wellenüberlagerung als Prozentanteil des Wellenzyklus

.pixelRadius Radius des Blur Filter-Raumes

Beispiel

```

<DIV STYLE="position:absolute; left:70px; filter:
progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true', ShadowOpacity=1.0)"
>
<IMG SRC="/test.gif" >
</DIV>

```

.positive Schattentransparenz

.Rotation Faktor der Rotation in 90-Grad-Schritten

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
<IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(rotation=3)'">
Rotate 270 degrees
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>

```

.shadowOpacity Schatten-Deckungskraft
wirkt nur wenn Eigenschaft makeShadow auf true gesetzt ist

Beispiel

```

<DIV STYLE="position:absolute; left:70px; filter:
progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true', ShadowOpacity=1.0)"
>
<IMG SRC="/test.gif" >
</DIV>

```



.sizingMethod Art der Anpassung
 nicht Filter Matrix

Beispiel

```
<SCRIPT>
    function FilterAendern(ArtDerAnpassung)
    { ID_Div.filters(0).sizingMethod= ArtDerAnpassung;}
</SCRIPT>
<DIV ID="ID_Div"
    STYLE="position:absolute; left:140px; height:150; width:200;
        filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
            SRC='test.jpg', sizingMethod='scale');"
    >
</DIV>
<BUTTON onclick=" FilterAendern('image');"> kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>
```

.SizingMethod Container-Resize ein/aus
 nur Filter Matrix

.slideStyle Art

Beispiel

```
<DIV STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Slide(
        duration=3, bands='8', slideStyle='PUSH');"
    >
```

.spokes Anzahl Speichen

.squaresX Anzahl der Spalten

Beispiel

```
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else // 0 entspricht false
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
        duration=5, direction='left', squaresX=4);"
    >
```

.squaresY Anzahl der Zeilen

Beispiel

```
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
```



```

    }
    else    // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
        duration=5, direction='left', squaresY=4);">

</DIV>

```

.SRC Url des Image

Beispiel

```

<SCRIPT>
    function FilterAendern(ArtDerAnpassung)
    { ID_Div.filters(0).sizingMethod= ArtDerAnpassung;}
</SCRIPT>
<DIV ID="ID_Div"
    STYLE="position:absolute; left:140px; height:150; width:200;
        filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
            SRC='test.jpg', sizingMethod='scale');"
    >

</DIV>
<BUTTON onclick=" FilterAendern('image');"> kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>

```

.StartColor Start-Deckungskraft als Integer

.StartColorStr Start-Deckungskraft als String

.startX Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft startet

.startY Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft startet

.status Status der Filteranimation

.strength Ausdehnung des Glühens

Beispiel

```

<SCRIPT>
    function FilterSetzen()
    {
        with (window.event.srcElement.filters[0])
        {
            if (strength < 200)
            {
                strength += 1;
                direction += 45;
            }
        }
    }
</SCRIPT>
<IMG SRC="test.gif"
    STYLE="filter:progid:DXImageTransform.Microsoft.motionBlur(STRENGTH=1, DIRECTION=0)"
    onfilterchange="FilterSetzen()"
>

```

.stretchStyle Art

Beispiel

```

<DIV STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Stretch(duration=2, stretchStyle='PUSH');"
>

```

.style Art der Deckungskraft



.transition

Filternummer

<u>numerischer Wert der Eigenschaft transition</u>	<u>alternative Kodierung ab IE 5.5</u>
0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE',
motion='in')	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE',
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
motion='out') 4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='right')
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
9 - horizontale Jalousie	DXImageTransform.Microsoft.CheckerBoard(direction='right')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.RandomDissolve
12 - zufällige Auflösung	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Strips(motion='leftdown')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftup')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='rightdown')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightup')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
22 - Streifen vertikal zufällig	
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Beispiel

```

<SCRIPT>
function go()
{
    ID_Span.filters[0].Apply();

    if (ID_Span.style.visibility == "visible")
    {
        ID_Span.style.visibility = "hidden";
        ID_Span.filters.revealTrans.transition=2;
    }
    else
    {
        ID_Span.style.visibility = "visible";
        ID_Span.filters[0].transition=3;
    }

    ID_Span.filters[0].Play();
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Play Transition" onClick="go();"></INPUT>
<SPAN ID="ID_Span" STYLE="position:absolute; visibility:visible;
    filter:revealTrans(DURATION=2, TRANSITION=3);
    width:300; height:300; background-color: lightgreen"
>
    <CENTER>
        <DIV STYLE="background-color:red;height=100;width:100;
            position:relative;top:100
            "
        >
        </DIV>
    </CENTER>
</SPAN>

```

.WipeStyle Richtung
nicht Filter RadialWipe

.wipeStyle Style des Wischens
nur Filter RadialWipe

.Xray Grauskale aus Mittelwert vom Rot- und Grünanteil ein/aus

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px; color:tan;" >
    <IMG SRC="/test.gif" >
</DIV>

```



```
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(XRay=1)'">
    Show Xray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter=''>Clear Filter</BUTTON>
```

.addAmbient()	indirektes Umgebungslicht (Ambiente)
.addCone()	direktes 3D-Licht als Lichttunnel oder Spot
.addPoint()	3D-Lichtpunkt als Lichtquelle, die in alle Richtungen strahlt
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart die Methode .play() benutzen (sichtbarmachen der Filtereinstellungen)
.changeColor()	Lichtfarbe nachträglich ändern
.changeStrength()	Lichtintensität nachträglich ändern
.clear()	alle Lichtquellen löschen
.moveLight()	Focus des Lichtkegels bzw. Lichtpunkt bewegen
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange



10. Anhang: Eigenschaften und Methoden des Windows Media Player 7.1

Eigenschaften:

.attributeCount	Anzahl der Attribute des aktuellen media Objektes
.attributeCount	Anzahl der Attribute der aktuellen Playliste
.autoStart	Autostart der Wiedergabe wenn Autostart erlaubt: Wiedergabe sofort gestartet bei Zuweisung von Werten zu ID_Player.URL ID_Player.currentMedia wenn Autostart nicht erlaubt ist: Wiedergabe nach Zuweisung von Werten zu ID_Player.URL ID_Player.currentMedia durch Aufruf von ID_Player.controls.play() Hinweis: Zuweisung zu ID_Player.URL ID_Player.currentMedia setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht
.balance	Stereo-Balance (Kanalausrichtung) -100 bis 100 Standard ist 0 -100 ganz links 100 ganz rechts 0 mittig
.bandWidth	aktuelle Bandbreite (Bits pro Sekunde) der Media-Datei mit Datenstrom also Video
.baseURL	Basis-Pfad einer HTTP-Url wird ignoriert von der Url-Angabe im Event ID_Player.scriptCommand (scType mit Wert "URL") relativer Pfad: "/" kodierbar, wenn nicht erstes Zeichen erstes Zeichen darf nicht sein "." "\" "/" wenn ja, so wird es automatisch gelöscht ".." etc. nicht kodierbar
.bitRate	aktuelle Bitrate der Media-Datei
.bufferingCount	Anzahl der Zeitpuffer während der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video
.bufferingProgress	aktueller Prozentanteil der Pufferung während der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video Fortgeschrittenheit der Pufferung 100% Pufferung entspricht komplett gepuffert
.bufferingTime	Anzahl der Millisekunden als Zeit für Pufferung vor der Wiedergabe der Media-Datei
.captioningID	Name des Frame oder des Control, in denen das Caption angezeigt wird
.cdromCollection	Zeiger auf Collection cdromCollection
.closedCaption	Zeiger auf Objekt closedCaption
.controls	Zeiger auf Objekt controls Hinweis: aktuelles media Objekt ID_Player.currentMedia aktuelles Media Item ID_Player.controls.currentItem aktuelle Playliste ID_Player.currentPlaylist Playliste ID_Player.playlistCollection.item()
.count	Anzahl der im System verfügbaren CD-Laufwerke Anzahl der Elemente in der Collection Achtung: nicht .length Anzahl der Media Item's in der aktuellen Playliste
.count	Anzahl der Elemente im Zeichenkettenfeld laut ID_Player.mediaCollection.getAttributeStringCollection()
.count	Anzahl der Playlisten in der Media-Bibliothek
.currentItem	Zeiger auf aktuelles media Objekt in einer Playliste (Media Item) media Objekt als aktuelles Element der Playliste setzen Nummer des aktuellen Marker in der Media-Datei Marker als aktuellen Marker setzen
.currentMarker	Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei Zeiger auf Objekt currentMedia für media Objekt und Media Item Hinweis: aktuelles media Objekt ID_Player.currentMedia aktuelles Media Item ID_Player.controls.currentItem aktuelle Playliste ID_Player.currentPlaylist Playliste ID_Player.playlistCollection.item()
.currentMedia	siehe ID_Player.URL Download und Wiedergabe parallel möglich bei Dateien mit Suffix *.ASF *.AVI *.MP3 *.MPEG



*.WAV
 *.WM
 *.WMA
 *.WMV

Autostart der Wiedergabe: laut ID_Player.settings.autoStart

wenn Autostart erlaubt:

Wiedergabe sofort gestartet bei Zuweisung von Werten zu

ID_Player.URL

ID_Player.currentMedia

wenn Autostart nicht erlaubt ist:

Wiedergabe nach Zuweisung von Werten zu

ID_Player.URL

ID_Player.currentMedia

durch Aufruf von ID_Player.controls.play()

Hinweis: Zuweisung zu ID_Player.URL

ID_Player.currentMedia

setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht

.currentPlaylist	<p>Zeiger auf Objekt currentPlaylist</p> <p>Hinweis: aktuelles media Objekt ID_Player.currentMedia</p> <p>aktuelles Media Item ID_Player.controls.currentItem</p> <p>aktuelle Playliste ID_Player.currentPlaylist</p> <p>Playliste ID_Player.playlistCollection.item()</p>
.currentPosition	<p>aktuelle Wiedergabe-Position als Wert in Sekunden in der Media-Datei ab Beginn der Datei</p> <p>aktuelle Wiedergabe-Position setzen</p>
.currentPositionString	aktuelle Wiedergabe-Position als String in der Media-Datei ab Beginn der Datei
.defaultFrame	<p>Standardname des Frame für Event ID_Player.scriptCommand, wenn scType mit Wert "URL",</p> <p>falls im Kommando-Wert kein anderer Frame kodiert wurde</p>
.downloadProgress	<p>aktueller Prozentanteil des Downloads des aktuellen media Objektes von einem Webserver</p> <p>Fortgeschrittenheit des Download</p> <p>100% Download entspricht komplett geladen</p> <p>Download und Wiedergabe parallel möglich bei Dateien mit Suffix</p> <p>*.ASF</p> <p>*.AVI</p> <p>*.MP3</p> <p>*.MPEG</p> <p>*.WAV</p> <p>*.WM</p> <p>*.WMA</p> <p>*.WMV</p>
.driveSpecifier	liefert Laufwerksbuchstabe des CD-Laufwerkes als Element der Collection
.duration	zeitliche Dauer in Sekunden (Integer) des aktuellen media Objektes
.durationString	zeitliche Dauer in Sekunden (String) des aktuellen media Objektes
.enableContextMenu	Kontextmenü ein bzw. aus bei rechte Maus-Click
.enabled	Windows Media Player-Control-Elemente ein bzw. aus (true bzw. false)
.enableErrorDialogs	automatische Fehleranzeige ein/ aus (true/false)
.encodedFrameRate	<p>Framerate (Frames pro Sekunden) der Media-Datei als Video</p> <p>Framerate laut Hersteller des Video</p> <p>nicht aktuelle Framerate des Video (siehe ID_Player.network.FrameRate)</p>
.error	Zeiger auf Objekt error
.errorCount	Anzahl der Fehlerinformationen in der Fehler-Queue
.errorDescription	Fehlertext (String)
.FrameRate	<p>aktuelle Framerate (Frames pro Sekunden) der Media-Datei als Video</p> <p>nicht Framerate laut Hersteller des Video (siehe ID_Player.network.encodedFrameRate)</p>
.framesSkipped	aktuelle Anzahl der bisher wiedergegebenen Frames ab Beginn der Wiedergabe des aktuellen media Objektes
.fullScreen	als Video
.fullScreen	Vollbildmodus bei Wiedergabe eines Video ein bzw. aus (true bzw. false)
.imageSourceHeight	Höhe in Pixel des aktuellen media Objektes
.imageSourceWidth	Breite in Pixel des aktuellen media Objektes
.invokeURLs	<p>Wirksamkeit der Url-Zuweisung an den Standard-Webbrowser aufgrund eines eintreffenden Events</p> <p>ID_Player.scriptCommand mit scType auf "URL" und Param mit Wert als Kette mit der zuzuweisenden Url</p> <p>Scriptkommando liegt z.B. in einer ASF-Datei</p> <p>Achtung: Wenn Script zum Event kodiert wurde, so wird dieses Script vor der Url-Zuweisung abgearbeitet. Das Script ist frei programmierbar und kann z.B. eine Wertänderung von Param und oder ID_Player.settings.invokeURLs durchführen.</p> <p>Danach wird erst die Url laut Wert von Param an den Webbrowser weitergeleitet, falls das ID_Player.settings.invokeURLs zulässt.</p> <p>Es muss kein Script zum Event ID_Player.scriptCommand zu scType mit Wert "URL" kodiert sein.</p> <p>true für weiterreichen</p>



	false für nicht weiterreichen
.isOnline	prüfen ob Client on- bzw. offline zum Netzwerk ist (z.B Internet) (true bzw. false)
.lostPackets	aktuelle Anzahl der bisher verlorengegangenen Datenpakete ab Beginn der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video Verlust an Datenpaketen nicht bei HTTP möglich hängt von der Netzwerkverbindung ab
.markerCount	Anzahl der Marker in der Media-Datei des aktuellen media Objektes Media-Datei muss Marker unterstützen Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei Nummer des aktuellen Marker laut ID_Player.controls.currentMarker
.maxBandwidth	maximale Bandbreite (Bits pro Sekunde) für die Wiedergaben der Media-Datei mit Datenstrom also Video besonders bei einem Datenfluss verenden, der mit verschiedenen Bitraten ausgestattet ist (MBS Multiple streams width different bit rates) z.B. Herabsetzung der maximalen Bitrate wegen der schlechteren Netzwerkqualität, weil Media-Datei mindestens eine höhere Bandbreite, als die Verbindung Server zum Client (Player) physisch leisten kann
.mediaCollection	Zeiger auf Collection mediaCollection Hinweis: aktuelles media Objekt ID_Player.currentMedia aktuelles Media Item ID_Player.controls.currentItem aktuelle Playliste ID_Player.currentPlaylist Playliste ID_Player.playlistCollection.item()
.mute	Stummschaltung für Audio bzw. Video mit Ton (true für stumm, false für nicht stumm)
.name	Name des aktuellen media Objektes
.name	Name der aktuellen Playliste Jede Playliste hat in der Media-Bibliothek einen eindeutigen Namen.
.network	Zeiger auf Objekt network
.openState	aktueller Status des Players bezüglich Playliste, Media-Datei, Codec, Lizenz, Individualisierung
.playCount	Anzahl der Wiedergaben (nicht der Wiederholungen) ab 1
.playlist	Zeiger auf Playliste auf einer CD im CD-Laufwerk aktuelle Playliste: siehe Objekt ID_Player.currentPlaylist
.playlistCollection	Zeiger auf Collection playlistCollection Hinweis: aktuelles media Objekt ID_Player.currentMedia aktuelles Media Item ID_Player.controls.currentItem aktuelle Playliste ID_Player.currentPlaylist Playliste ID_Player.playlistCollection.item()
.playState	Status der Wiedergabe der aktuellen Media-Datei (Status des Players bei Wiedergabe) Media-Datei muss bestimmte Wiedergabe-Möglichkeiten unterstützen, damit deren Status angezeigt wird Download und Wiedergabe parallel möglich bei Dateien mit Suffix *.ASF *.AVI *.MP3 *.MPEG *.WAV *.WM *.WMA *.WMV
	Integer 0 undefined 1 Wiedergabe ist gestoppt 2 Wiedergabe pausiert 3 Wiedergabe erfolgt fortlaufend 4 Wiedergabe mit schnellem Vorlauf 5 Wiedergabe mit schnellem Rücklauf 8 Wiedergabe ist komplett und geendet 10 Wiedergabe kann beginnen, da Player bereit dafür ist
.rate	Faktor für Wiedergaberate laut Media-Datei Veränderung der Wiedergabe: reale Wiedergaberate = Wiedergaberate laut Medium * Faktor Floating point immer 1 bei Audio < 0 und > 0 bei Medium mit Datenstrom bei ASF oder WMV < 0 für Rückwärtswiedergabe > 0 für Vorwärtswiedergabe > 0 und bei Medium, das nicht Audio und nicht ASF und nicht WMV < 1 so geringere Rate als laut Medium > 1 so höhere Rate als laut Medium Standard ist 1.0 (Rate laut Medium) aber bei ID_Player.controls.fastForward 5.0 ID_Player.controls.fastReverse -5.0
.receivedPackets	aktuelle Anzahl der bisher nicht verlorengegangenen Datenpakete ab Beginn der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video Verlust an Datenpaketen nicht bei HTTP möglich



	hängt von der Netzwerkverbindung ab
.receptionQuality	aktuelle Prozentzahl der in den letzten 30 Sekunden nicht verlorengegangenen Datenpakete während der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video, gegenüber der Gesamtanzahl aller in den letzten 30 Sekunden von der Media-Datei abgesendeten Datenpakete, also inklusive der verlorenen Datenpakete.
	Verlust an Datenpaketen nicht bei HTTP möglich
	hängt von der Netzwerkverbindung ab
.recoveredPackets	aktuelle Anzahl der bisher verlorengegangenen aber wiederhergestellten Datenpakete ab Beginn der Wiedergabe des aktuellen media Objektes mit Datenstrom, also Video
	Verlust an Datenpaketen nicht bei HTTP möglich
	hängt von der Netzwerkverbindung ab
.SAMIFileName	Name der Synchronized Accessible Media Interchange (SAMI)-Datei, die Informationen zum Caption enthält
	Suffix der Datei *.smi oder *.sami
.SAMILang	Sprache der der Synchronized Accessible Media Interchange (SAMI)-Datei, die Informationen zum Caption enthält
	Suffix der Datei *.smi oder *.sami
	Sprache liegt in der Datei innerhalb <STYLE> .. </STYLE>
	z.B. .ENUSCC für US-Englisch
	Sprache ist alphanumerisch und muss mit Punkt beginnen
	mehrere Sprachen möglich
.SAMISStyle	Style des Caption
	Sprache liegt in der Datei innerhalb <STYLE> .. </STYLE>
	ist Kette mit erstem Zeichen stets das Nummernkreuz #
	Bsp. #BigFont
.settings	Zeiger auf Objekt settings
.sourceProtocol	Datentransport-Protokoll während der Wiedergabe des media Objektes
.sourceURL	Url des aktuellen media Objektes
.status	interner und laufender Status des Players
.stretchToFit	Playerfenster bei Media-Datei als Video in der Ansicht automatisch auf Videogröße ausdehnen
	ein bzw. aus
	sinnvoll, wenn Video größere Dimension haben könnte als das Layout des Players (Player-Fenster)
	true, so Playerfenster automatisch anpassen auf Videogröße
	false Default
	Playerfenster nicht automatisch anpassen auf Videogröße
.uiMode	Umfang der anzuzeigenden Control-Elemente des Players aus der Menge aller Control-Elemente zum aktuellen Media-Datei-Typ
	Control-Elemente kann der User zur Steuerung der Wiedergabe benutzen
	Control-Elemente nur sichtbar, wenn
	der Player sichtbar ist
	falls im OBJECT-Tag das HEIGHT-Attribut kodiert wurde, dann mit Wert > 40 Pixel
	Syntax
	String
	"none" keine Control-Elemente anzeigen
	User kann nichts steuern
	Achtung: Im OBJECT-Tag stets HEIGHT=0 und WIDTH=0 setzen
	"mini" angezeigt werden nur die Elemente
	Pause-Taste
	Stop-Taste
	Mute-Taste
	Lautstärkeregelung
	"mini" angezeigt werden alle Elemente
.URL	Url der Media-Datei, die wiedergegeben werden soll
	Media-Datei wird geladen und zum aktuellen Medium (aktuelles media Objekt)
	Autostart der Wiedergabe: laut ID_Player.settings.autoStart
	wenn Autostart erlaubt:
	Wiedergabe sofort gestartet bei Zuweisung von Werten zu
	ID_Player.URL
	ID_Player.currentMedia
	wenn Autostart nicht erlaubt ist:
	Wiedergabe nach Zuweisung von Werten zu
	ID_Player.URL
	ID_Player.currentMedia
	durch Aufruf von ID_Player.controls.play()
	Hinweis: Zuweisung zu
	ID_Player.URL
	ID_Player.currentMedia
	setzt Medium immer als aktuelles Medium, egal ob Autostart erlaubt oder nicht
	siehe ID_Player.currentMedia
.versionInfo	Version der instanziierten Windows Media Player
.volume	Lautstärke für Ton
	Achtung: Verändert den Regler zur Media-Datei-Art in der Windows-Lautstärke-Regelung und nicht den Master-Regler !



Bps.: Wiedergabe einer MID-Datei und Volume-Veränderung, so
 Regler der MID-Datei verändert
 Unbedingt pro Mediumart vor der Volume-Veränderung den aktuellen Wert von
 Volume per Script sichern und nach der Wiedergabe rückspeichern !

Integer

0 bis 100

0 für stumm

100 für maximal

Standard laut aktueller Windows Lautstärke-Regelung zum Medium-Typ

Methoden:

.add()	Media-Datei laden und als media Objekt der Bibliothek hinzufügen und Zeiger liefern
.appendItem()	media Objekt an das Ende der aktuellen Playliste anhängen, also dort Media-Item erzeugen aktuelle Playliste erweitern
.attributeName()	Name eines Attributes der aktuellen Playliste liefern
.clearErrorQueue()	Fehler-Queue löschen, also alle Fehlerinformationen zu allen media Objekten / Media Item's löschen
.close()	schliesst den Player im HTML-Dokument
.eject()	CD-Laufwerk als Element der Collection öffnen ohne Laufwerksbuchstabe
.fastForward()	Achtung: CD darf nicht gerade wiedergegeben werden, also vorher ID_Player.playState prüfen schnelles Vorspulen während der Wiedergabe einer Media-Datei Standardgeschwindigkeit des schnellen Vorspulens während der Wiedergabe ist 5 faches der normalen Wiedergabegeschwindigkeit ansonsten per ID_Player.setting.rate einstellbar
	ignoriert ID_Player.settings.rate
	wird aufgehoben durch ID_Player.controls.play
	ID_Player.controls.stop
	hebt auf ID_Player.controls.fastReverse()
	nicht möglich z.B. für Audio
	Ermittlung, ob Medium schnelles Vorspulen bei Wiedergabe unterstützt, per ID_Player.controls.isAvailable()
.fastReverse()	schnelles Rückspulen während der Wiedergabe einer Media-Datei Standardgeschwindigkeit des schnellen Rückspulens während der Wiedergabe ist 5 faches der normalen Wiedergabegeschwindigkeit ansonsten per ID_Player.setting.rate einstellbar
	nur innerhalb eines Track
	ignoriert ID_Player.settings.playCount
	ID_Player.settings.rate
	wird aufgehoben durch ID_Player.controls.fastForward()
	ID_Player.controls.play
	ID_Player.controls.stop
	nicht möglich z.B. für Audio
	Ermittlung, ob Medium schnelles Rückspulen bei Wiedergabe unterstützt, per ID_Player.controls.isAvailable()
.getAll()	Playlist aus allen Elemente der Media-Bibliothek erzeugen und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getAll()	Zeiger auf Feld aller Playlisten in der Media-Bibliothek liefern
.getAttributeName()	Name eines Attributes des aktuellen media Objektes liefern
.getAttributeStringCollection()	Zeichenkettenfeld aus allen Werte eines Attributes zu einem Media-Typ erzeugen pro media Objekt vom Media-Typ wird ein Element in der Collection erzeugt media Objekt nur dann verwendet, wenn auch Attribut mit ihm verbunden ist
	Bsp.: für Attribut:
	"Album"
	"Author"
	Bsp.: für Media-Typ
	"Audio"
	"Video"
.getByAlbum()	Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Album" und darin einen bestimmten Wert besitzen, und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getByAttribute()	Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die ein Attribut mit bestimmten Wert besitzen, und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getByAuthor()	Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Author" und darin einen bestimmten Wert besitzen, und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getByDriveSpecifier()	Zeiger auf ein CD-Laufwerk als Element der Collection liefern: mit Laufwerksbuchstabe
.getByGenre()	Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Genre" und darin einen bestimmten Wert besitzen, und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getByName()	Playliste aus all denjenigen Elementen der Media-Bibliothek erzeugen, die das Attribut "Name" und darin einen bestimmten Wert besitzen, und Zeiger liefern
	je mehr Elemente in der Bibliothek, um so länger dauert die Abarbeitung der Methode
.getByName()	Zeiger auf Feld aus maximal 1 Playliste aus der Media-Bibliothek liefern



	jede Playliste hat einen eindeutigen Namen: Feld kann also maximal 1 Element besitzen siehe auch .getAll()
.getItemInfo()	Wert eines Attributes des aktuellen media Objektes liefern
.getItemInfo()	Wert eines Attributes eines Media Item's in der aktuellen Playliste liefern
.getItemInfoByAtom()	Wert eines Attributes des aktuellen media Objektes liefern
.getMarkerName()	Name eines Markers in der Media-Datei des aktuellen media Objektes liefern Media-Datei muss Marker unterstützen Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount Nummer des aktuellen Marker laut ID_Player.controls.currentMarker
.getMarkerTime()	Zeitpunkt eines Markers in der Media-Datei des aktuellen media Objektes liefern Media-Datei muss Marker unterstützen Marker dient der Wiedergabe der Media-Datei ab diesem Punkt in der Media-Datei Anzahl der Marker in der Media-Datei laut ID_Player.currentMedia.markerCount Nummer des aktuellen Marker laut ID_Player.controls.currentMarker
.getMediaAtom()	Index eines Attributes liefern, also Art des Attributes
.getMode()	Modus der Wiedergabe von Tracks (Modus des Players) ermitteln (zufällig, endlos) siehe Event ID_Player.modeChange
.getProxyBypassForLocal()	prüfen, ob Proxyserver übergangen wird, wenn der Server als Media-Datei-Quelle im lokalen Netzwerk liegt
.getProxyExceptionList()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Ausnahmeliste des Proxyservers liefern bezüglich Computer Domains IP's die den Proxyserver übergehen
.getProxyName()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Name des benutzten Proxyservers liefern
.getProxyPort()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Nummer des benutzten Ports des Proxyservers liefern
.getProxySettings()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Einstellungen des Proxyservers liefern
.importPlaylist()	Playliste in die Media-Bibliothek importieren
.insertItem()	media Objekt in die aktuelle Playliste einfügen, also dort Media Item erzeugen aktuelle Playliste erweitern vor dem Einfügen: Media Item am Einfügeplatz und seine Nachfolger rutschen automatisch 1 Platz weiter runter in der Playliste es wird ID_Player.currentPlaylist.count erhöht um Wert 1
.isAvailable()	Verfügbarkeit von Controls zu einer Media-Datei
.isAvailable()	Veränderbarkeit von ID_Player.settings.rate prüfen
.isDeleted()	prüfen ob Media Datei aus dem Datei-Pool der Bibliothek entfernt wurde bei lokaler Media Datei: den Papierkorb des Systems prüfen bei nicht-lokaler Datei: auf logische Verknüpfung prüfen Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen: ID_Player.mediaCollection.setDeleted() Media-Datei aus ID_Player.mediaCollection. entfernen und optional aus der Datenbank der Namen: ID_Player.mediaCollection.remove()
.isDeleted()	prüfen ob Playliste im Papierkorb des Systems liegt
.isIdentical()	auf Identität des aktuellen media Objektes mit einem anderen (nicht aktuellen) media Objektes prüfen
.isIdentical()	auf Identität der aktuellen Playliste mit einer anderen (nicht aktuellen) Playliste prüfen
.isMemberOf()	prüfen ob aktuelles media Objekt eine Media Item ist, also einer Playliste angehört
.isReadOnlyItem()	Veränderbarkeit eines Attributes des aktuellen media Objektes
.item()	Zeiger auf ein CD-Laufwerk als Element der Collection liefern: ohne Laufwerksbuchstabe
.item()	Zeiger auf ein Media Item in der aktuellen Playliste liefern per Zeiger kann Media Item wie ein media Objekt behandelt werden
.item()	Zeiger auf Fehler (errorItem Objekt) in der Fehler-Queue liefern
.item()	Wert eines Elementes im Zeichenkettenfeld laut ID_Player.mediaCollection.getAttributeStringCollection() ermitteln
.item()	Zeiger auf eine Playliste in der Media-Bibliothek
.launchURL()	eine Media-Datei an den Standardbrowser im System schicken
.moveItem()	Media Item in der aktuellen Playliste verschieben durch Indextausch aktuelle Playliste verändern
.newPlaylist()	Playliste mit Name als Objekt instanzieren und Zeiger liefern, aber nicht in die Media-Bibliothek importieren Jede Playliste muss einen eindeutigen Namen haben: Es wird Fehler erzeugt, wenn Name bereits in der Media-Bibliothek vorhanden ist.
.next()	nächsten Eintrag (Media Item) vorwärts in der Playliste anwählen: Wenn bereits LETZTER Eintrag der Playliste erreicht wurde und dann .next() aktiviert wird, dann wird der ERSTE Eintrag in der Playliste eingestellt. Medium muss Playliste unterstützen
.pause()	pausieren der Wiedergabe des aktuellen media Objektes bzw. Media Items siehe auch ID_Player.controls.play() ID_Player.controls.playItem()



	ID_Player.controls.stop()
.play()	Wiedergabe des aktuellen media Objektes bzw. Media Item starten oder wenn aktuelles Media bzw. Media Item bereits pausiert, so wird Pause aufgehoben siehe auch ID_Player.controls.playItem() ID_Player.controls.pause() ID_Player.controls.stop()
.playItem()	Laden eines Playlist-Eintrages (Media Item) und Autostart der Wiedergabe ID_Player.setting.autoStart wird ignoriert Hinweis: beliebige Media Datei laden und wiedergeben per per ID_Player.url und danach ID_Player.controls.play() siehe auch ID_Player.controls.play() ID_Player.controls.pause() ID_Player.controls.stop()
.previous()	vorhergehenden Eintrag (Media Item) rückwärts in der Playliste anwählen: Wenn bereits ERSTER Eintrag der Playliste erreicht wurde und dann .previous() aktiviert wird, dann wird der LETZTE Eintrag in der Playliste eingestellt.
.remove()	Medium muss Playliste unterstützen media Objekt aus Media-Bibliothek, also aus ID_Player.mediaCollection, entfernen: aus Datenbank der Namen der Objekte und optional aus dem Datei-Pool Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen: ID_Player.mediaCollection.setDeleted()
.remove()	Playliste aus der Media-Bibliothek entfernen
.removeItem()	Media Item aus der aktuellen Playliste entfernen aktuelle Playliste bereinigen Wird das zu entfernende Media Item gerade wiedergegeben, dann wird Wdergabe gestoppt und das nächste Playlisten-Eement aktiviert. Wenn nächstes Item nicht vorhanden, so Vorgänger vom entfernten Item aktiviert. Wenn auch das nicht vorhanden ist, also aktuelle Playliste leer ist, dann wird ID_Player.currentMedia auf null-Zeiger gesetzt !
.setDeleted()	Media-Datei aus dem Datei-Pool der Media-Bibliothek entfernen: bei lokaler Media Datei: immer in den Papierkorb des Systems verschieben bei nicht-lokaler Datei: logische Verknüpfung entfernen
.setDeleted()	Playliste aus der Media-Bibliothek in den Papierkorb des System verschieben
.setItemInfo()	Wert eines Attributes des aktuellen media Objektes setzen wenn Attribut nicht vorhanden, so erzeugt Veränderbarkeit eines Attributes laut ID_Player.currentMedia.isReadOnlyItem()
.setItemInfo()	Wert eines Attributes eines Media-Item's in der aktuellen Playliste setzen wenn Attribute nicht vorhanden, so erzeugt es besteht kein Schreibschutz für Attribute
.setMode()	Modus der Wiedergabe von Tracks (Modus des Players) setzen (zufällig, endlos) siehe Event ID_Player.modeChange
.setProxyBypassForLocal()	einstellen, ob Proxyserver übergangen wird, wenn der Server als Media-Datei-Quelle im lokalen Netzwerk liegt
.setProxyExceptionList()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Ausnahmeliste des Proxyservers setzen bezüglich Computer Domains IP's die den Proxyserver übergehen
.setProxyName()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Name des benutzten Proxyservers setzen
.setProxyPort()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Nummer des benutzten Ports des Proxyservers setzen
.setProxySettings()	nur möglich, wenn ID_Player.network.getProxySettings() den Wert 2 liefert Einstellungen des Proxyservers setzen
.stop()	Wiedergabe des aktuellen media Objektes bzw. Media Item (aktuelles media Objekt der Playliste) beenden oder wenn aktuelles Media bzw. Media Item bereits pausiert, so wird Pause aufgehoben Achtung: Die Systemressourcen zur Media-Datei werden freigegeben ! bei Track: automatisch immer auf Trackanfang gesetzt siehe auch ID_Player.controls.play() ID_Player.controls.playItem() ID_Player.controls.pause()
.webHelp()	Hilfe-Seite von Microsoft im Internet zum Windows Media Player aktivieren Auf der Hilfe-Seite stehen Informationen zu Fehlern.



11 . Index

- 75
 ' 56, 57, 58, 113, 114, 1101
 'entwerten 56, 58, 114
 ! 75, 1101
 != 75, 76
 !== Operator 78
 !DOCTYPE 424, 812, 822, 1595, 1937
 !DOCTYPE HTML PUBLIC 802
 !DOCTYPE 801
 " 56, 57, 58, 113, 114, 1101
 " 56
 " 57
 " 58
 " 113
 " 114
 " 1101
 "entwerten 56, 58, 114
 "#default#behaviorName 249, 482, 563, 568, 573, 591,
 599, 602, 610, 616, 619, 628, 649, 660, 663, 667, 682,
 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731,
 737, 740, 745, 750, 751, 754, 761, 767, 776, 1019,
 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072,
 1079, 1189, 1192, 1196, 1737
 "fr" 1659
 "javascript:document.write('Hallo!');" 41
 # 241, 306, 333, 449, 561, 565, 572, 734, 1101, 1166,
 1167, 1631
 #clientCaps 852
 #default 1761
 #default#clientCaps 852
 #default#download 855
 #default#homePage 856
 #default#httpFolder 857
 #default#mediaBar 858
 #default#saveFavorite 862
 #default#saveHistory 863
 #default#saveSnapshot 864
 #default#time2 876
 #default#userdata 975
 #download 855
 #format_name{eigenschaften_liste} 977
 #hashtext 1166
 #homePage 856
 #httpFolder 857
 #mediaBar 858
 #rrggb 980, 987, 990, 991, 992
 #saveFavorite 862
 #saveHistory 863
 #saveSnapshot 864
 #text 244, 561, 567, 572, 590, 598, 601, 614, 619, 627,
 649, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708,
 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766,
 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058,
 1064, 1070, 1077, 1188, 1191, 1661
 #time2 876
 #userdata 975
 \$ 1101
 \$zeichenfolge 1183
 % 75, 1101
 % Operator 78
 %= 76
 %20 144, 211
 & 75, 76, 144, 1101
 && 75, 419
 &{javascript_ausdruck}; 41
 &b 419
 &b&b 419
 &d 419
 &D 419
 816, 826, 1948
 &p 419
 &P 419
 &t 419
 &T 419
 &u 419
 &w 419
 (1101
 () 75
 (*.class 450, 566, 570
 (bedingung) ? wert1 : wert2 66
 (suchmuster_element) 1184
) 1101
 * 75, 1101
 * / % << >> >>> & ^ 77
 *.ANI 337, 781
 *.asf 875, 917, 930, 1131, 1469, 2000
 *.ASF 874, 928, 972, 1468, 1497
 *.asx 875, 917, 930, 949, 974, 1131, 1470, 1999
 *.ASX 934
 *.avi 671
 *.AVI 874, 1468
 *.bmp 671
 *.cdf 1164, 1830
 *.chm 405, 1901
 *.class-Datei 567, 753, 1594
 *.css 996, 1634
 *.CUR 337, 781
 *.emf 671
 *.eot 981
 *.gif 508, 671
 *.htc 492, 504, 507, 809, 819, 850, 865, 1761, 1919
 *.htc-Datei 479, 1147
 *.htm 405, 1901
 *.icm 511, 2001
 *.jpeg 671
 *.jpg 671
 *.js 45, 46
 *.LGX 1509
 *.MID 874, 1468
 *.mov 671
 *.MP3 874, 1468
 *.mpeg 671
 *.MPEG 874, 1468
 *.mpg 671
 *.png 671
 *.prf 981
 *.sami 1484
 *.smi 1484
 *.TXT 1509
 *.WAV 874, 1468
 *.wax 875, 1470
 *.WM 874, 1468
 *.wma 875, 1469
 *.WMA 874, 1468
 *.wmf 671
 *.wmv 875, 1469
 *.WMV 874, 1468
 *.wvx 875, 1470



*.xbm 671
 *= 76
 , 1101
 . 1101
 .& 1187
 .* 1187
 ._ 1187
 .` 1187
 .' 1187
 .+ 1187
 .\$01 bis .\$09 1187
 .\$1 bis .\$9 1187
 .0 161, 1553
 .above 327
 .abs() 199, 1734
 .AbsolutePosition 1379
 .abstract 907, 913, 920, 925, 936, 944, 969, 1553
 .accelerate 885, 892, 893, 896, 897, 903, 904, 907, 908,
 913, 914, 918, 920, 921, 925, 926, 931, 944, 945, 954,
 964, 965, 969, 970, 1555, 1606
 .accept 686, 1555
 .acceptCharset 626, 1556
 .accessKey 481, 560, 567, 571, 589, 597, 609, 614, 618,
 680, 686, 691, 695, 703, 707, 712, 717, 721, 725, 730,
 743, 752, 759, 774, 1014, 1027, 1032, 1041, 1046,
 1051, 1057, 1063, 1069, 1076, 1190, 1556
 .accumulate 892, 893, 896, 897, 903, 904, 949, 964, 965,
 1556, 1562, 1568, 1584
 .acos() 199, 1734
 .action 626, 1558
 .activeDur 889, 1559
 .activeElement 1560
 .activeTime 889, 1560
 .activeTimeToParentTime() 886, 894, 898, 905, 909,
 915, 919, 922, 928, 932, 946, 950, 955, 966, 972, 1734
 .activeTimeToSegmentTime() 886, 894, 898, 905, 909,
 915, 919, 922, 928, 932, 946, 950, 955, 966, 972, 1736
 .activeTrack 939, 1561
 .add 511, 2001
 .add() 455, 456, 457, 507, 742, 761, 769, 771, 1198,
 1736
 .Add() 1427, 1442, 1737
 .addAmbient() 518, 2008
 .addBehavior() 239, 249, 482, 563, 568, 573, 591, 599,
 602, 610, 615, 619, 628, 649, 660, 663, 667, 682, 688,
 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737,
 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029,
 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079,
 1189, 1192, 1196, 1737
 .AddChannel() 1162, 1737
 .addComponentRequest() 855, 1737
 .addCone() 518, 2008
 .AddDesktop() 1162, 1737
 .AddFavorite() 1162, 1738
 .addImport() 996, 1738
 .additive 893, 897, 903, 904, 949, 965, 1557, 1562,
 1568, 1584, 1666
 .addPageRule() 996, 1738
 .addPoint() 518, 2008
 .addReadRequest() 1172, 1738
 .addRule() 996, 999, 1738
 .alert() 388, 1738
 .align 567, 609, 614, 666, 680, 686, 703, 752, 759, 1014,
 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069,
 1562
 .aLink 589, 1564

.alinkColor 424, 1564
 .all 1196
 .allowTransparency 648, 666, 1564
 .alt 567, 571, 680, 686, 703, 752, 1564
 .altHTML 567, 752, 1565
 .altKey 1104, 1106, 1153, 1154, 1565
 .altLeft 1104, 1106, 1153, 1154, 1565
 .anchor() 214, 1738
 .anchors 1197
 .appName 410, 1167, 1169, 1566
 .appendChild() 241, 250, 591, 599, 602, 610, 616, 619,
 628, 660, 663, 667, 688, 693, 697, 705, 709, 714, 718,
 723, 727, 732, 740, 745, 767, 776, 1019, 1029, 1034,
 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1192,
 1739
 .AppendData 1378
 .appendData() 281, 602, 1739
 .applets 1197
 .apply() 195, 518, 1739, 2008
 .Apply(); 528, 672
 .applyElement() 240, 241, 250, 482, 563, 568, 573, 591,
 599, 602, 610, 616, 620, 629, 649, 660, 663, 668, 682,
 688, 693, 697, 701, 705, 710, 714, 719, 723, 727, 732,
 737, 740, 745, 754, 761, 767, 776, 1019, 1029, 1034,
 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189,
 1192, 1739
 .appMinorVersion 410, 1169, 1566
 .appName 37, 410, 1167, 1169, 1566, 1567
 .appVersion 1167, 1169, 1567
 .appVersion 410
 .appVersion 1567
 .archive 567, 752, 1567
 .areas 1198
 .arguments 194, 1567
 .asin() 199, 1739
 .assign() 1167, 1739
 .assing(url) 1166
 .atan() 199, 1739
 .atan2() 199, 1739
 .atEnd() 189, 1739
 .AtEndOfLine 1443, 1568
 .AtEndOfStream 1443, 1568
 .attachEvent() 388, 393, 430, 433, 482, 483, 507, 563,
 568, 569, 573, 574, 591, 599, 602, 610, 611, 616, 620,
 629, 649, 650, 660, 663, 664, 668, 683, 688, 693, 697,
 698, 701, 705, 710, 714, 715, 719, 723, 727, 728, 732,
 735, 737, 740, 745, 746, 754, 755, 761, 767, 776, 777,
 1019, 1029, 1030, 1034, 1038, 1043, 1048, 1049, 1053,
 1054, 1060, 1066, 1072, 1079, 1107, 1147, 1189, 1193,
 1740, 1741, 1759
 .attributeName 893, 897, 904, 965, 1562
 .attributeCount 859, 860, 861, 862, 1568, 1778, 1793,
 1854, 1855, 1907
 .attributeName 893, 897, 949, 1568
 .attributes 1198
 .Attributes 1437, 1439, 1570
 .attributName 892, 896, 903, 964, 1556
 .author 907, 913, 920, 925, 936, 944, 969, 1570
 .autocomplete 627, 708, 725, 1572
 .AutoCompleteSaveForm() 1163, 1742
 .autoReverse 885, 887, 893, 895, 897, 900, 904, 907,
 909, 911, 913, 915, 917, 918, 920, 922, 924, 925, 928,
 930, 931, 934, 944, 946, 949, 951, 954, 957, 965, 967,
 970, 972, 974, 1128, 1573, 1690, 1725, 1996
 .AutoScan() 1163, 1742
 .AvailableSpace 1434, 1574



.availHeight 854, 1182, 1574
 .availLeft 1182
 .availTop 1182
 .availWidth 854, 1182, 1574
 .avi 702
 .back() 1165, 1166, 1742
 .background 327, 589, 1014, 1027, 1041, 1063, 1069, 1574
 .balance 581, 1574
 .bands 511, 2001
 .Banner 913, 925, 940, 944, 970, 1575
 .BannerAbstract 913, 926, 940, 944, 970, 1576
 .BannerMoreInfo 913, 926, 940, 944, 970, 1578
 .BaseHref 752, 1579
 .begin 560, 571, 597, 609, 614, 618, 627, 666, 681, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 743, 765, 774, 872, 885, 887, 893, 895, 897, 900, 904, 906, 907, 909, 911, 913, 915, 917, 920, 922, 924, 926, 928, 930, 934, 936, 944, 946, 949, 951, 954, 957, 965, 970, 972, 974, 1014, 1027, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1127, 1191, 1579, 1725, 1994
 .beginElement() 878, 886, 890, 894, 898, 905, 909, 915, 919, 923, 928, 932, 946, 950, 955, 966, 972, 1742
 .beginElementAt() 886, 894, 898, 905, 910, 916, 923, 928, 932, 946, 950, 955, 966, 972, 1743
 .behavior 743, 1581
 .behaviorUrns 1198
 .below 328
 .bgColor 328, 424, 589, 743, 809, 818, 1014, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069, 1581, 1918
 .bgProperties 589, 1581
 .Bias 511, 2001
 .big() 214, 1744
 .blink() 215, 1744
 .blockDirection 481, 589, 609, 614, 627, 849, 1581
 .blockFormats 1199
 .blur() 389, 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 633, 634, 635, 637, 638, 641, 649, 660, 668, 683, 688, 693, 697, 705, 710, 714, 719, 723, 727, 732, 735, 745, 754, 761, 776, 1019, 1030, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1193, 1745
 .bmp 702
 .BOF 1379
 .bold() 215, 1745
 .Boolean() 143, 152, 1745
 .border 659, 666, 671, 681, 752, 1014, 1581
 .borderColor 648, 659, 1014, 1057, 1058, 1063, 1069, 1581
 .borderColorDark 1014, 1058, 1063, 1069, 1581
 .borderColorLight 1014, 1058, 1063, 1069, 1581
 .bottom 462, 1087, 1581
 .bottomMargin 589, 1582
 .boundElements 1199
 .boundingHeight 459, 1083, 1582
 .boundingLeft 459, 1083, 1582
 .boundingTop 459, 1083, 1583
 .boundingWidth 459, 1083, 1583
 .browserLanguage 411, 1167, 1169, 1583
 .bufferDepth 854, 1182, 1583
 .bufferingProgress 907, 913, 920, 926, 944, 970, 1583
 .BuildPath() 1431, 1745
 .button 1104, 1107, 1158, 1584
 .by 892, 893, 897, 900, 903, 904, 949, 964, 965, 1556, 1562, 1568, 1584, 1586

.calcMode 892, 893, 894, 897, 898, 903, 904, 905, 949, 964, 965, 1556, 1568, 1584, 1586, 1588, 1647, 1648, 1670, 1728
 .call() 195, 1745
 .callee 161, 1589
 .caller 195, 1589
 .cancelBubble 1104, 1107, 1589, 1772
 .canHaveChildren 240, 241, 242, 481, 560, 597, 601, 614, 618, 627, 659, 662, 666, 686, 691, 695, 703, 708, 712, 717, 721, 725, 730, 739, 743, 752, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1069, 1076, 1191, 1589
 .canHaveHTML 240
 .canPause 907, 913, 920, 926, 944, 970, 1590
 .canSeek 886, 895, 899, 906, 907, 910, 913, 916, 919, 921, 923, 926, 929, 933, 944, 947, 948, 950, 951, 956, 967, 970, 973, 1590, 1877, 1878, 1879, 1880
 .caption 1014, 1590
 .captureEvent() 329
 .captureEvents() 291, 293, 298, 300, 359, 360, 361, 1144, 1145, 1146
 .CaseSensitive 1378
 .ceil() 199, 1745
 .cellIndex 1063, 1069, 1590
 .cellPadding 1014, 1590
 .cells 1199
 .cellSpacing 1014, 1591
 .changeColor() 518, 2008
 .changeStrength() 518, 2008
 .charAt() 215, 1745
 .charCodeAt() 216, 1746
 .charset 424, 560, 736, 749, 1187, 1591
 .CharSet 1378
 .checked 634, 636, 691, 712, 1591
 .childNodes 1199
 .children 1200
 .class-Datei 567, 753, 1594
 .classid 752, 1591
 .className 481, 560, 567, 571, 589, 597, 609, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 700, 703, 708, 712, 717, 721, 725, 730, 739, 743, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1191, 1591
 .clear() 431, 518, 770, 1746, 2008
 .clearAttributes() 238, 240, 250, 483, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 697, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 754, 761, 767, 776, 826, 1019, 1030, 1034, 1038, 1043, 1048, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1746
 .clearComponentRequest() 855, 1746
 .clearData() 416, 1112, 1746
 .clearInterval() 390, 402, 1747
 .clearRequest() 1172, 1748
 .clearTimeout() 391, 404, 1748
 .click() 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 633, 634, 637, 638, 683, 688, 693, 698, 705, 710, 715, 719, 723, 727, 732, 740, 746, 754, 761, 767, 776, 1019, 1030, 1043, 1048, 1054, 1060, 1066, 1072, 1079, 1193, 1748
 .clientHeight 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 .clientInformation 378, 1592



.clientLeft 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 .clientTop 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 .clientWidth 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 .clientX 1104, 1107, 1158, 1592
 .clientY 1104, 1107, 1158, 1592
 .clip 328
 .clip.bottom 328
 .clip.height 328
 .clip.left 328
 .clip.right 328
 .clip.top 328
 .clip.width 328
 .clip='rect(x1,y1,x2,y2)' 993
 .clipBegin 908, 913, 921, 926, 944, 970, 1593
 .clipboardData 378, 1593
 .clipBottom 849, 1593
 .clipLeft 849, 1593
 .clipRight 849, 1594
 .clipTop 849, 1594
 .cloneNode() 239, 251, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1749
 .close() 391, 431, 440, 1749, 1750
 .Close() 1444, 1751
 .closed 378, 1594
 .code 567, 753, 1594
 .codeBase 567, 753, 1594
 .codeType 753, 1594
 .collapse() 459, 1083, 1751
 .color 511, 618, 1594, 2001
 .colorDepth 854, 1182, 1594, 1595
 .colorSpace 511, 2001
 .cols 659, 1014, 1076, 1595
 .colSpan 1063, 1070, 1595
 .Column 1443, 1595
 .compareEndPoints() 459, 1083, 1751
 .compareVersions() 855, 1751
 .compatMode 424, 801, 1595
 .compile() 1185
 .complete 671, 681, 686, 703, 1595
 .componentFromPoint() 483, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 750, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751
 .concat() 166, 216, 1752, 1753, 1859
 .confirm() 392, 1753
 .connectionType 411, 854, 1595, 1665
 .constructor 61, 150, 209, 1595, 1596

.contains() 251, 483, 563, 569, 574, 591, 599, 611, 616, 620, 629, 650, 660, 664, 668, 683, 688, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 750, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1753
 .content 749, 1596, 1683
 .contentEditable 481, 508, 560, 589, 597, 609, 614, 618, 627, 686, 708, 713, 717, 721, 725, 730, 743, 774, 797, 849, 1076, 1191, 1596, 1631
 .contentWindow 648, 666, 1597
 .controlRange 1200
 .cookie 424, 1597
 .cookieEnabled 411, 854, 1167, 1170, 1598
 .coords 560, 571, 1598
 .Copy() 1438, 1441, 1753
 .CopyFile() 1431, 1753
 .CopyFolder() 1431, 1753
 .copyright 908, 914, 921, 926, 936, 944, 970, 1599
 .cos() 199, 1753
 .Count 1199, 1427, 1436, 1441, 1442, 1601
 .cpuClass 411, 854, 1170, 1601
 .createAttribute() 223, 238, 251, 431, 484, 592, 600, 603, 612, 617, 621, 762, 768, 778, 826, 1753
 .createCaption() 1019, 1754
 .createComment() 238, 252, 432, 1754
 .createControlRange() 457, 458, 459, 591, 770, 771, 1083, 1723, 1754, 1757, 1883
 .createDocumentFragment() 419, 432, 1754
 .createElement() 222, 238, 240, 241, 252, 262, 268, 271, 432, 455, 456, 457, 468, 471, 507, 587, 593, 601, 603, 612, 617, 621, 630, 662, 665, 669, 690, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 742, 747, 761, 763, 769, 771, 778, 826, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1068, 1074, 1081, 1194, 1736, 1754, 1870
 .createEventObject() 433, 499, 1107, 1755
 .CreateFolder() 1431, 1757
 .createPopup() 392, 1178, 1757
 .createRange() 457, 458, 459, 770, 771, 1083, 1723, 1757, 1883
 .createRangeCollection() 458, 770, 1083, 1757
 .createStyleSheet() 239, 241, 253, 433, 826, 1757
 .CreateTextFile() 1431, 1757
 .createTextNode() 239, 242, 253, 280, 433, 1757
 .createTextRange() 457, 458, 459, 770, 771, 1083, 1723, 1757, 1883
 .createTFOot() 1019, 1758
 .createTHead() 1019, 1758
 .cssText 464, 1601
 .ctrlKey 1104, 1107, 1153, 1154, 1601
 .ctrlLeft 1104, 1107, 1153, 1154, 1602
 .current 1165
 .currentFrame 908, 914, 921, 926, 945, 970, 1603
 .currentItem 858, 1603
 .currentStyle.hasLayout 797
 .data 601, 753, 1603
 .data[] 1104
 .dataFld 560, 567, 597, 609, 614, 648, 666, 681, 686, 691, 700, 703, 708, 713, 725, 730, 743, 759, 774, 1076, 1108, 1603
 .dataFormatAs 597, 609, 686, 730, 743, 774, 1108, 1604
 .dataPageSize 1014, 1604
 .dataSrc 560, 567, 597, 609, 648, 666, 681, 686, 691, 700, 703, 708, 713, 725, 730, 743, 759, 774, 1016, 1076, 1108, 1605
 .dataTransfer 1104



.DateCreated 1437, 1439, 1605
 .DateLastAccessed 1437, 1439, 1606
 .DateLastModified 1437, 1439, 1606
 .decelerate 885, 892, 893, 896, 897, 903, 904, 907, 908, 913, 914, 918, 920, 921, 925, 926, 931, 944, 945, 954, 964, 965, 969, 970, 1555, 1606
 .declare 753, 1606
 .decodeURI() 143, 152, 1758
 .decodeURIComponent() 143, 152, 1758
 .defaultCharset 425, 1606
 .defaultChecked 634, 636, 691, 713, 1606
 .defaultSelected 765, 1607
 .defaultStatus 378, 1607
 .defaultValue 635, 641, 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 1076, 1607
 .defer 1187, 1607
 .Delete() 1438, 1441, 1758
 .deleteCaption() 1019, 1758
 .deleteCell() 1060, 1758
 .deleteData() 280, 602, 1759
 .DeleteFile() 1432, 1759
 .DeleteFolder() 1432, 1759
 .deleteRow() 1019, 1043, 1049, 1054, 1759
 .deleteTFoot() 1019, 1043, 1759
 .deleteTHead() 1019, 1043, 1759
 .description 193, 1168, 1169, 1171, 1607, 1656
 .designMode 425, 995, 1607
 .detachEvent() 388, 393, 430, 433, 483, 507, 563, 568, 569, 573, 574, 591, 599, 602, 610, 611, 616, 620, 629, 649, 650, 660, 663, 664, 668, 683, 688, 693, 697, 698, 701, 705, 710, 714, 715, 719, 723, 727, 728, 732, 735, 737, 740, 745, 746, 754, 755, 761, 767, 776, 777, 1019, 1030, 1034, 1038, 1043, 1048, 1049, 1053, 1054, 1060, 1066, 1072, 1079, 1107, 1147, 1189, 1193, 1740, 1741, 1759
 .deviceXDPI 1183, 1607
 .deviceYDPI 1183, 1608
 .dialogArguments 378, 1608
 .dialogHeight 379, 381, 1609, 1610
 .dialogLeft 380, 1609, 1610
 .dir 425, 481, 561, 571, 589, 597, 609, 614, 618, 627, 662, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 739, 743, 753, 759, 765, 774, 1016, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1064, 1070, 1077, 1191, 1610
 .direction 511, 743, 1610, 2001
 .Direction 512, 2001
 .disabled 481, 561, 572, 582, 589, 597, 601, 609, 614, 618, 627, 648, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 743, 749, 753, 759, 765, 774, 1016, 1028, 1032, 1058, 1064, 1077, 1188, 1191, 1610, 1635
 .disabledUI 859, 1613
 .disableExternalCapture() 293
 .doComponentRequest() 855, 1760
 .doctype 425, 1614
 .document 381, 1181, 1614
 .document.all 1201
 .documentElement 238, 239, 240, 241, 243, 425, 1616
 .documentTimeToParentTime() 886, 894, 899, 905, 910, 916, 919, 923, 928, 932, 947, 950, 955, 966, 972, 1760
 .doImport() 507, 1761
 .domain 426, 1616
 .doReadRequest() 1172, 1763
 .downloadCurrent 908, 914, 921, 926, 945, 970, 1616
 .downloadTotal 908, 914, 921, 926, 945, 970, 1616

.dragDrop() 563, 574, 591, 599, 602, 611, 620, 629, 650, 664, 668, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 740, 746, 750, 755, 761, 767, 777, 1019, 1030, 1034, 1060, 1066, 1080, 1189, 1763
 .Drive 1437, 1439, 1617
 .DriveExists() 1432, 1764
 .DriveLetter 1434, 1617
 .DriveType 1434, 1617
 .dropEffect 1112, 1617, 1620
 .duplicate() 459, 1084, 1764
 .dur 609, 872, 885, 893, 897, 898, 904, 908, 909, 914, 915, 918, 921, 922, 926, 927, 928, 931, 932, 936, 940, 945, 946, 949, 954, 965, 966, 971, 972, 1619, 1680, 1725
 .duration 512, 860, 1620, 2001
 .Dx 512, 2001
 .Dy 512, 2001
 .dynsrc 681, 686, 703, 1620
 .E 199, 1620
 .effectAllowed 1112, 1617, 1620
 .elementFromPoint() 434, 1764
 .elements 1201
 .embeds 1201
 .emf 702
 .empty() 770, 1764
 .enabled 512, 859, 860, 1119, 1130, 1621, 1986, 1998, 2001
 .enabledPlugin 451, 613, 756, 1171
 .enableExternalCapture() 283, 293, 361, 1146
 .enablePlugin 1169, 1171
 .encodeURI() 143, 152, 1758, 1764
 .encodeURIComponent() 144
 .encodeURIComponent() 153, 1764
 .enctype 627, 1621
 .end 561, 572, 597, 609, 614, 618, 627, 666, 681, 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 743, 765, 774, 872, 885, 887, 893, 895, 897, 900, 904, 906, 908, 909, 911, 914, 915, 917, 919, 921, 922, 924, 926, 928, 929, 934, 937, 945, 946, 948, 949, 951, 954, 956, 965, 967, 971, 972, 973, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1117, 1191, 1621, 1725, 1984
 .EndColor 512, 2002
 .EndColorStr 512, 2002
 .endElement() 878, 886, 887, 890, 894, 895, 899, 900, 905, 906, 910, 911, 916, 917, 919, 923, 924, 928, 929, 932, 934, 947, 948, 950, 951, 955, 956, 966, 967, 972, 973, 1117, 1764, 1984
 .endElementAt() 886, 894, 899, 905, 910, 916, 923, 928, 932, 947, 950, 955, 966, 972, 1765
 .endSync 881, 918, 931, 1623
 .EOF 1379
 .escape() 144, 152, 153, 211, 1764, 1766
 .EscapeChar 1378
 .eval() 43, 144, 145, 153, 154, 211, 394, 1766, 1769
 .event 381, 1188, 1623, 1634
 .exec() 218, 1185, 1869
 .execCommand() 276, 434, 457, 459, 476, 771, 1084, 1201, 1768
 .execScript() 394, 1769
 .Exists() 1427, 1769
 .exp() 199, 1769
 .expand() 459, 1084, 1770
 .expando 426, 465, 1624
 .expires 976, 1624
 .expression() 240, 241, 253, 826, 1770
 .external 381, 1625



.face 618, 1625
 .fgColor 426, 1625
 .FieldDelim 1378
 .fileCreatedDate 426, 681, 1625
 .FileExists() 1432, 1770
 .fileModifiedDate 427, 681, 1625
 .filename 1168
 .Files 1440, 1625
 .fileSize 427, 681, 1625
 .FileSystem 1434, 1625
 .fileUpdatedDate 681, 1626
 .fill 885, 887, 893, 895, 897, 900, 904, 906, 908, 909,
 911, 914, 915, 917, 919, 921, 922, 924, 926, 928, 929,
 931, 933, 945, 946, 948, 949, 951, 954, 956, 967, 971,
 972, 973, 1117, 1626, 1725, 1984
 .filter 510
 .Filter 1378
 .filters 1201
 .filters[0].Apply(); 528, 672
 .filters[0].Play(); 528, 672
 .FilterType 512, 2002
 .findText() 459, 1084, 1770
 .finishOpacity 512, 2002
 .finishX 512, 2002
 .finishY 512, 2002
 .fire() 501, 1770
 .fireChange() 502, 1772
 .fireEvent() 483, 563, 569, 574, 591, 599, 602, 611, 616,
 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701,
 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 751,
 755, 761, 767, 777, 997, 1019, 1030, 1034, 1038,
 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1107, 1189,
 1193, 1196, 1772
 .firstChild 241, 243, 561, 567, 572, 597, 601, 614, 618,
 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708,
 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775,
 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064,
 1070, 1077, 1188, 1191, 1627
 .firstPage() 1019, 1773
 .fixed() 216, 1774
 .floor() 199, 1775
 .focus() 394, 435, 483, 563, 569, 574, 591, 599, 611,
 616, 620, 629, 633, 634, 635, 637, 638, 641, 650, 661,
 668, 683, 689, 690, 693, 694, 698, 699, 705, 707, 710,
 711, 715, 716, 719, 720, 723, 725, 728, 729, 732, 735,
 746, 755, 761, 777, 1021, 1030, 1043, 1049, 1054,
 1060, 1066, 1072, 1080, 1193, 1775, 1883
 .FolderExists() 1432, 1775
 .fontcolor() 216, 1775
 .fonts 1202
 .fontsize() 216, 1775
 .fontSmoothingEnabled 1183, 1627
 .form 597, 614, 633, 634, 635, 636, 637, 638, 641, 686,
 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 753,
 759, 765, 1077, 1627
 .forms 1202
 .forward() 1165, 1166, 1775
 .frame 1016, 1627
 .frameBorder 648, 659, 666, 1627
 .frameElement 381, 1627
 .frames 382, 1202, 1627
 .frameSpacing 659, 1627
 .FreeSpace 1434, 1627
 .freq 512, 2002
 .from 893, 898, 900, 904, 965, 1628, 1629
 .fromCharCode() 217, 1775

.fromElement 1105, 1107, 1159, 1630
 .function 512, 2002
 .galleryImg 681, 1630
 .GetAbsolutePathName() 1432, 1776
 .getAdjacentText() 242, 254, 483, 563, 569, 574, 591,
 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683,
 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732,
 737, 740, 746, 755, 761, 767, 777, 1021, 1030, 1034,
 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1189,
 1193, 1776
 .getAttribute() 238, 240, 254, 483, 563, 569, 574, 591,
 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683,
 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732,
 737, 740, 746, 750, 755, 761, 768, 777, 827, 849, 850,
 863, 864, 1021, 1030, 1034, 1038, 1044, 1049, 1054,
 1060, 1066, 1072, 1080, 1172, 1189, 1193, 1777, 1778
 .getAttributeName() 859, 860, 861, 862, 1568, 1793,
 1854, 1855, 1907
 .getAttributeNode() 238, 255, 483, 563, 569, 574, 591,
 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683,
 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732,
 737, 741, 746, 751, 755, 761, 768, 777, 1021, 1030,
 1034, 1038, 1044, 1049, 1054, 1060, 1066, 1072, 1080,
 1189, 1193, 1196, 1778
 .GetBaseName() 1432, 1778
 .getBookmark() 459, 1084, 1770, 1778
 .getBoundingClientRect() 460, 483, 564, 569, 574, 591,
 600, 602, 611, 616, 620, 629, 683, 689, 693, 698, 705,
 710, 715, 719, 723, 728, 732, 735, 737, 741, 746, 755,
 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049,
 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1778
 .getClientRects() 460, 483, 564, 569, 574, 591, 600, 602,
 611, 616, 620, 629, 683, 689, 693, 698, 705, 710, 715,
 719, 723, 728, 732, 737, 741, 746, 755, 762, 768, 777,
 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067,
 1072, 1080, 1084, 1193, 1780
 .getComponentVersion() 855, 1781
 .getData() 416, 1104, 1112, 1781
 .getDate() 180, 1781
 .getDay() 180, 1783
 .GetDrive() 1432, 1785
 .GetDriveName() 1433, 1786
 .getElementById() 239, 255, 435, 827, 1786
 .getElementsByName() 239, 256, 435, 436, 483, 564,
 569, 574, 592, 600, 611, 617, 620, 629, 650, 661, 664,
 668, 683, 732, 737, 741, 746, 762, 777, 827, 1021,
 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1073,
 1081, 1189, 1193, 1786, 1787
 .getElementsByTagName() 239, 255, 256, 435, 436,
 483, 564, 569, 574, 592, 600, 611, 616, 620, 629, 650,
 661, 664, 668, 683, 732, 737, 741, 746, 762, 777, 827,
 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067,
 1073, 1080, 1189, 1193, 1786
 .getExpression() 240, 241, 253, 257, 483, 564, 574, 592,
 600, 611, 617, 620, 629, 668, 683, 689, 693, 698, 701,
 706, 710, 715, 719, 724, 728, 733, 746, 755, 762, 768,
 777, 826, 827, 849, 850, 1021, 1030, 1034, 1038,
 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1193, 1770,
 1787
 .GetExtensionName() 1433, 1788
 .GetFile() 1433, 1788
 .GetFileName() 1433, 1788
 .GetFolder() 1433, 1788
 .getFullYear() 180, 1788
 .getHours() 180, 1791
 .GetInterfaceVersion() 1513



.getItemInfo() 859, 860, 861, 862, 1568, 1778, 1793, 1854, 1855, 1907
 .getMilliseconds() 180, 1793
 .getMinutes() 181, 1795
 .getMonth() 181, 1797
 .getNamedItem() 267, 466, 1198, 1799
 .GetParentFolderName() 1433, 1800
 .getSeconds() 181, 1800
 .GetSpecialFolder() 1433, 1802
 .GetTempName() 1433, 1802
 .getTime() 181, 1802
 .getTimezoneOffset() 181, 1804
 .getUTCDate() 181, 1807
 .getUTCDay() 181, 1809
 .getUTCFullYear() 181, 1811
 .getUTCHours() 182, 1813
 .getUTCMilliseconds() 182, 1815
 .getUTCMinutes() 182, 1818
 .getUTCMonth() 182, 1820
 .getUTCSeconds() 182, 1822
 .GetVersion() 1514
 .getYear() 182, 1824
 .gif 702
 .global 1185
 .go() 1166, 1824
 .go(position) 1165
 .gradientSize 512, 2002
 .GradientType 512, 2002
 .grayScale 512, 2002
 .gridSizeX 512, 2002
 .gridSizeY 512, 2002
 .handleEvent() 294, 329, 360, 735, 1145
 .hasAudio 908, 914, 921, 926, 945, 971, 1630
 .hasChildNodes() 240, 242, 258, 564, 569, 574, 592, 600, 602, 611, 617, 621, 630, 650, 661, 664, 668, 683, 689, 694, 698, 706, 710, 715, 719, 724, 728, 733, 737, 741, 746, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1189, 1193, 1824
 .hasDownloadProgress 908, 914, 921, 926, 945, 971, 1630
 .hasFeature() 239, 1825
 .hasFocus() 437, 611, 1825
 .hash 306, 449, 561, 565, 572, 734, 1166, 1631
 .hasLayout 797, 849, 1631
 .hasMedia 561, 572, 597, 609, 614, 618, 627, 666, 681, 686, 691, 696, 700, 703, 708, 713, 717, 721, 726, 744, 765, 775, 885, 893, 898, 904, 908, 914, 921, 927, 937, 945, 949, 954, 959, 971, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1191, 1631
 .hasNextItem 859, 1631
 .hasOwnProperty() 150, 155, 209, 211, 1677, 1825
 .hasPlaylist 914, 927, 945, 971, 1632
 .hasVisual 908, 914, 921, 927, 945, 971, 1632
 .height 328, 648, 666, 671, 681, 744, 753, 854, 1016, 1058, 1064, 1070, 1105, 1159, 1182, 1183, 1632
 .hide() 1181, 1825
 .hideFocus 481, 561, 567, 572, 589, 597, 609, 614, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 744, 753, 759, 775, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1191, 1632
 .higher 943, 1633
 .hight 803
 .history 382, 1633
 .host 561, 572, 734, 1166, 1167, 1633

.hostname 561, 572, 734, 1166, 1167, 1633
 .href 306, 449, 559, 561, 562, 565, 572, 734, 736, 996, 1166, 1167, 1631, 1633, 1634, 1685
 .hreflang 561, 736, 1634
 .hspace 567, 666, 681, 686, 703, 744, 753, 1634
 .htaccess und .htpasswd und Passwortschutz 51
 .htaccess und Passwortschutz 51
 .htc-Datei 504, 506, 507, 1761
 .htmlFor 730, 1188, 1623, 1634
 .htmlText 459, 1083, 1634
 .htpasswd und Passwortschutz 51
 .httpEquiv 749, 750, 1596, 1634
 .ico 420, 1634, 1679
 .id 481, 561, 567, 572, 582, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 735, 736, 739, 744, 750, 751, 753, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 .ignoreCase 1185
 .images 1203
 .implementation 427, 1635
 .ImportExportFavorites() 1164, 1827
 .imports 1203
 .indeterminate 691, 1078, 1606, 1635, 1699
 .index 636, 765, 937, 1635, 1636
 .indexOf() 217, 1827
 .innerHTML 481, 561, 589, 597, 609, 614, 618, 627, 659, 662, 665, 730, 739, 744, 759, 766, 775, 1000, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1188, 1191, 1636
 .innerText 481, 561, 589, 593, 597, 609, 614, 618, 627, 662, 666, 730, 735, 739, 744, 759, 766, 775, 1000, 1016, 1028, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1637
 .input 1187
 .inRange() 460, 1084, 1828
 .insertAdjacentElement() 239, 258, 564, 569, 574, 592, 600, 602, 611, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 698, 701, 706, 710, 715, 719, 724, 728, 733, 738, 741, 746, 755, 762, 768, 777, 1021, 1030, 1035, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1189, 1194, 1829
 .insertAdjacentHTML() 240, 241, 258, 455, 484, 564, 574, 592, 600, 603, 612, 617, 621, 630, 661, 669, 684, 689, 694, 698, 701, 706, 711, 715, 720, 724, 728, 733, 741, 742, 746, 762, 768, 777, 827, 1031, 1067, 1073, 1081, 1194, 1829
 .insertAdjacentText() 242, 258, 484, 564, 574, 592, 600, 603, 612, 617, 621, 630, 661, 669, 684, 689, 694, 698, 701, 706, 711, 715, 720, 724, 728, 733, 741, 746, 762, 768, 777, 827, 1031, 1067, 1073, 1081, 1194, 1829
 .insertBefore() 241, 259, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 698, 706, 711, 715, 720, 724, 728, 733, 741, 747, 755, 762, 768, 777, 1021, 1031, 1035, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1194, 1829
 .insertCell() 1061, 1830
 .insertData() 281, 603, 1830
 .insertRow() 1022, 1044, 1049, 1055, 1830
 .intent 512, 2002
 .invert 512, 2002
 .irisStyle 513, 2002
 .isActive 886, 889, 894, 895, 898, 899, 905, 906, 909, 910, 915, 916, 919, 922, 923, 928, 929, 932, 933, 937, 939, 946, 947, 948, 950, 951, 955, 956, 966, 967, 972,



973, 1561, 1734, 1736, 1742, 1743, 1764, 1765, 1850, 1853, 1873, 1877, 1878, 1879, 1880, 1884
.isComponentInstalled() 855, 1830
.isContentEditable 482, 561, 567, 572, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1640
.isDisabled 482, 561, 567, 572, 582, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1640
.isEqual() 460, 1084, 1830
.isFinite() 146, 155, 205, 1830
.isHomePage() 857, 1830
.isLoading() 1514
.isMap 681, 1641
.isMultiLine 482, 561, 567, 572, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 750, 751, 753, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1641
.isMuted 889, 1641
.isMuting() 1514
.isNaN() 146, 156, 206, 1830
.isOn 889, 1642
.isOpen 1181, 1643
.isPause() 1514
.isPaused 890, 1645
.isPrototypeOf() 156, 212, 1830
.IsReady 1434, 1646
.IsRootFolder 1440, 1646
.isSpeaking() 1515
.isStreamed 908, 914, 921, 927, 945, 971, 1646
.IsSubscribed() 1164, 1830
.isTextEdit 482, 561, 567, 572, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 750, 753, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1647
.italics() 217, 1831
.item() 190, 267, 271, 274, 448, 450, 451, 452, 453, 454, 455, 456, 457, 461, 466, 470, 473, 504, 507, 511, 566, 571, 593, 613, 630, 641, 642, 652, 685, 707, 738, 742, 757, 769, 771, 890, 940, 998, 999, 1000, 1040, 1046, 1085, 1108, 1171, 1190, 1197, 1831
.Item() 1428, 1436, 1441, 1442, 1832, 1833
.Items() 1428, 1833
.javaEnabled 412, 854, 1647, 1834
.javaEnabled() 412, 1168, 1170, 1834
.join() 167, 1834
.jpeg 702
.jpg 702
.Key() 1428, 1834
.keyCode 1100, 1105, 1107, 1153, 1154, 1647
.Keys() 1429, 1835
.keySpline 965
.keySplines 893, 894, 898, 900, 904, 905, 1647, 1648, 1728
.keyTimes 893, 894, 898, 900, 904, 905, 965, 1647, 1648, 1728
.label 766, 1649

.lang 482, 561, 567, 572, 589, 597, 601, 609, 614, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
.language 482, 561, 567, 572, 590, 597, 609, 614, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 759, 766, 775, 1017, 1028, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1187, 1188, 1191, 1649, 1723
.Language 1378
.lastChild 241, 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
.lastIndex 1185
.lastIndexOf() 217, 1835
.lastMatch 1187
.lastModified 427, 1649
.lastPage() 1022, 1835
.lastParent 1187
.latestMediaTime 908, 914, 921, 927, 945, 971, 1649
.layerX 1105, 1106, 1159
.layerY 1105, 1159
.left 328, 462, 1087, 1649
.leftContext 1187
.leftMargin 590, 1650
.length 162, 166, 195, 214, 267, 271, 274, 275, 311, 382, 448, 450, 451, 452, 453, 454, 455, 456, 457, 461, 465, 470, 473, 474, 490, 504, 507, 511, 566, 571, 593, 601, 613, 641, 642, 652, 685, 707, 738, 742, 757, 759, 769, 771, 890, 940, 998, 999, 1000, 1040, 1046, 1085, 1108, 1165, 1168, 1169, 1171, 1190, 1650, 1651
.lightStrength 513, 2002
.Line 1444, 1651
.link 590, 1651
.link() 218, 1837
.linkColor 427, 1651
.links 1203
.LN10 199, 1651
.LN2 199, 1651
.load() 330, 976, 1837
.LoadText() 1515
.localeCompare() 220, 1838
.location 382, 427, 1651
.location.hash 306, 449, 565
.location.href 306, 449, 565
.location.reload() 222
.log() 199, 1838
.LOG10E 199, 1652
.LOG2E 199, 1652
.logicalXDPI 1183, 1652
.logicalYDPI 1183, 1652
.longDesc 648, 666, 681, 1652
.loop 582, 681, 703, 744, 747, 1653, 1906
.lowsrsrc 671, 681, 686, 703, 1654
.M11 513, 2002
.M12 513, 2002
.M21 513, 2002
.M22 513, 2002
.makeShadow 513, 2002
.marginHeight 648, 666, 1654
.marginWidth 648, 666, 1654
.Mask 513, 2003
.MaskColor 513, 2003



.match() 218, 1838
 .max() 199, 1838
 .maxLength 708, 726, 1654
 .maxSquare 513, 2003
 .media 798, 809, 818, 1655
 .mediaDur 908, 914, 921, 927, 931, 945, 954, 971, 1655
 .mediaHeight 908, 914, 921, 927, 931, 945, 954, 971, 1655
 .menuArguments 1162, 1656
 .mergeAttributes() 238, 240, 241, 259, 437, 484, 564, 569, 574, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 698, 702, 706, 711, 715, 720, 724, 728, 733, 738, 741, 747, 755, 762, 768, 777, 827, 1023, 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081, 1189, 1194, 1838
 .message 193, 1607, 1656
 .method 627, 1656
 .Methods 561, 1656
 .mimeType 908, 914, 921, 927, 945, 971, 1656
 .mimeTypes 1170, 1657
 .mimeTypes[] 1167, 1168
 .min() 199, 1839
 .mirror 513, 2003
 .mode 965, 1657
 .modifiers 1105, 1156
 .motion 513, 514, 2003
 .mov 702
 .move() 460, 1084, 1839
 .Move() 1438, 1441, 1839
 .moveAbove() 330
 .moveBelow() 330
 .moveBy() 330, 394, 1839
 .moveEnd() 460, 1084, 1839
 .MoveFile() 1433, 1839
 .moveFirst() 190, 1840
 .MoveFolder() 1433, 1840
 .moveLight() 518, 2008
 .moveNext() 191, 1840
 .moveRow() 1023, 1044, 1050, 1055, 1841
 .moveStart() 460, 1084, 1842
 .moveTo() 330, 395, 1842
 .moveToAbsolute() 330
 .moveToBookmark() 459, 460, 1084, 1770, 1778, 1842
 .moveToElementText() 460, 1084, 1842
 .moveToPoint() 460, 1084, 1842
 .mpeg 702
 .mpg 702
 .multiline 1187
 .multiple 759, 1658, 1723
 .mute 885, 908, 914, 921, 927, 932, 945, 954, 971, 1658
 .n 161, 1553
 .name 194, 328, 382, 507, 559, 561, 567, 597, 627, 633, 634, 635, 636, 637, 638, 641, 648, 659, 666, 671, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 736, 739, 749, 750, 753, 760, 860, 861, 1077, 1168, 1596, 1659, 1721
 .Name 1438, 1440, 1660
 .namedItem() 448, 450, 451, 452, 453, 454, 455, 456, 457, 461, 511, 566, 571, 614, 630, 641, 642, 652, 685, 707, 738, 742, 757, 770, 771, 998, 1000, 1040, 1046, 1085, 1171, 1190, 1197, 1842
 .namedRecordset() 569, 755, 1196, 1842
 .nameProp 681, 1660
 .namespaces 1203
 .navigate() 395, 857, 1844
 .NavigateAndFind() 1164, 1844

.navigateFrame() 858, 1845
 .navigateHomePage() 857, 1845
 .navigator 382, 1660
 .navigator.appName 37, 1567
 .next 1165
 .nextElement() 955, 1845
 .nextItem 859, 1660
 .nextPage() 1023, 1845
 .nextSibling 243, 561, 567, 572, 590, 598, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1660
 .nextSibling() 241
 .nextTrack() 928, 932, 940, 947, 955, 1847
 .nodeName 241, 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661
 .nodeType 239, 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661
 .nodeValue 238, 239, 242, 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661
 .noHref 572, 1662
 .noResize 649, 1662
 .normalize() 222, 239, 259, 484, 564, 569, 574, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 762, 768, 777, 1025, 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1196, 1848
 .noWrap 590, 609, 1064, 1070, 1662
 .number 194, 1662
 .Number() 146, 156, 206, 1848
 .object 567, 753, 1662
 .offsetParent 459, 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 682, 687, 692, 696, 704, 708, 713, 717, 718, 722, 726, 731, 739, 740, 744, 753, 775, 1017, 1028, 1033, 1036, 1037, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1083, 1192, 1664
 .offsetParent 382, 1662
 .offsetLeft 482, 561, 567, 572, 598, 601, 610, 619, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1191, 1663
 .offsetHeight 482, 561, 567, 572, 590, 598, 609, 615, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 735, 739, 744, 753, 760, 766, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1191, 1662
 .offsetHeight 482, 561, 567, 572, 598, 601, 610, 619, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1191, 1663
 .offsetLeft 459, 482, 561, 567, 572, 590, 598, 610, 615, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713,



717, 722, 726, 731, 735, 739, 744, 753, 760, 766, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1083, 1191, 1663

.offsetParent 459, 482, 561, 567, 572, 590, 598, 601, 609, 610, 615, 619, 627, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 760, 766, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1058, 1059, 1064, 1070, 1077, 1083, 1191, 1662, 1663

.offsetTop 459, 482, 561, 567, 572, 590, 598, 601, 610, 615, 619, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 735, 739, 744, 753, 760, 766, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1071, 1077, 1083, 1191, 1663, 1664

.offsetWidth 482, 561, 567, 572, 590, 598, 601, 610, 615, 619, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 735, 739, 744, 753, 760, 766, 775, 1017, 1028, 1033, 1036, 1037, 1042, 1047, 1052, 1059, 1064, 1070, 1071, 1077, 1191, 1192, 1663, 1664

.offsetX 1105, 1107, 1159, 1665

.offsetY 1105, 1107, 1159, 1665

.offX 514, 2003

.offY 514, 2003

.onLine 411, 1170, 1665

.onload 855

.onOffBehavior 482, 561, 590, 598, 610, 615, 628, 663, 667, 682, 687, 692, 696, 704, 708, 713, 718, 722, 726, 731, 744, 760, 766, 775, 849, 850, 1017, 1029, 1033, 1037, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1192, 1665

.opacity 514, 2003

.open() 395, 437, 1709, 1749, 1849

.open(["mime-typ"] [, "replace"]) 437

.OpenAsTextStream() 1438, 1850

.opener 221, 382, 395, 438, 1666, 1849

.openState 859, 860, 1124, 1666, 1991

.OpenTextFile() 1433, 1850

.options 760, 1204, 1666

.orientation 514, 2004

.origin 904, 1666

.outerHTML 482, 561, 567, 572, 598, 610, 615, 619, 628, 659, 663, 665, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1029, 1033, 1037, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1666

.outerText 482, 561, 568, 572, 598, 601, 610, 615, 619, 628, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1667

.overlap 514, 2004

.ownerDocument 238, 239, 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1667

.owningElement 996, 1667

.pages 1204

.pageX 328, 1105, 1159

.pageY 328, 1105, 1159

.parent 221, 383, 1667

.parentElement 240, 245, 482, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033,

1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1195, 1667

.parentElement() 460, 1084, 1850

.ParentFolder 1438, 1440, 1668

.parentLayer 329

.parentNode 240, 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668

.parentStyleSheet 996, 1668

.parentTextEdit 240, 245, 482, 562, 568, 572, 590, 598, 602, 615, 619, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668

.parentTimeBegin 890, 1669

.parentTimeEnd 890, 1669

.parentTimeToActiveTime() 886, 894, 899, 905, 910, 916, 919, 923, 929, 933, 947, 950, 955, 966, 972, 1850

.parentTimeToDocumentTime() 886, 894, 899, 905, 910, 916, 919, 923, 929, 933, 947, 950, 955, 966, 973, 1851

.parentWindow 428, 1670

.parse() 182, 1851

.parseFloat() 147, 157, 206, 411, 1567, 1851

.parseInt() 147, 157, 207, 1852

.pasteHTML() 460, 827, 1084, 1853

.path 893, 897, 898, 904, 1584, 1628, 1670

.Path 1434, 1438, 1440, 1671

.pathname 562, 572, 1167, 1671

.pathname 735, 1166

.pause() 592, 1853

.pauseElement() 886, 894, 899, 905, 910, 916, 923, 929, 933, 947, 950, 955, 966, 973, 1853

.PauseSpeaking() 1515

.Percent 514, 2004

.phase 514, 2004

.PI 199, 1672

.pixelDepth 1182

.pixelRadius 514, 2004

.platform 411, 854, 1167, 1170, 1672

.play() 518, 2008

.Play(); 528, 672

.player 908, 914, 921, 927, 945, 971, 1672

.playerObject 909, 914, 922, 927, 945, 971, 1674

.playlistInfo 859, 1675

.playNext() 859, 860, 1675, 1854, 1855, 1907

.playState 859, 860, 861, 862, 1568, 1675, 1778, 1793, 1854, 1855, 1907

.playURL() 859, 860, 1675, 1854, 1855, 1907

.plugins 1170, 1204, 1675

.png 703

.pop() 170, 172, 1855, 1899

.port 562, 572, 735, 1166, 1167, 1675

.position 992

.positive 514, 2004

.pow() 199, 1855

.preference() 1168

.prevElement() 955, 1855

.previous 1165

.previousPage() 1025, 1856

.previousSibling 241, 246, 562, 568, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744,



753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042,
1048, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1675
.prevTrack() 929, 933, 940, 947, 955, 1857
.print() 398, 584, 1859
.progress 890, 1676
.prompt() 398, 1859
.propertyIsEnumerable 150, 209, 1676
.propertyName 1105, 1107, 1676
.protocol 428, 562, 572, 682, 735, 1166, 1167, 1675,
1677
.prototype 111, 150, 155, 156, 160, 209, 1677, 1825,
1830
.prototype 70
.pseudoClass 998, 1678
.push() 166, 170, 173, 1752, 1859
.queryCommandEnabled() 279, 441, 457, 460, 479, 771,
1085, 1201, 1859
.queryCommandIndeterm() 279, 441, 457, 460, 479,
771, 1085, 1201, 1859
.queryCommandState() 279, 441, 457, 460, 479, 771,
1085, 1201, 1859
.queryCommandSupported() 280, 441, 457, 460, 479,
771, 1085, 1201, 1859
.queryCommandValue() 280, 441, 457, 460, 479, 771,
1085, 1201, 1859
.random() 199, 1859
.rating 909, 915, 922, 927, 937, 945, 971, 1678
.Read() 1444, 1860
.ReadAll() 1444, 1860
.ReadLine() 1444, 1860
.readOnly 709, 726, 996, 1077, 1678, 1679
.readyState 239, 246, 428, 482, 507, 562, 568, 572, 583,
590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682,
687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731,
736, 740, 744, 750, 751, 753, 760, 766, 775, 1017,
1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071,
1078, 1188, 1192, 1195, 1679
.recalc() 441, 827, 1860
.RecordCount 1379
.recordNumber 562, 598, 610, 649, 667, 682, 692, 700,
709, 713, 726, 731, 744, 760, 775, 1078, 1108, 1679
.recordset 753, 1195, 1378, 1679
.recordset() 1379
.referrer 428, 1679
.refresh() 1026, 1029, 1168, 1727, 1865
.rel 561, 562, 572, 736, 1167, 1633, 1679
.releaseCapture() 446, 484, 564, 569, 575, 592, 600, 612,
617, 621, 630, 684, 689, 694, 699, 702, 706, 711, 716,
720, 724, 728, 733, 735, 741, 747, 755, 762, 768, 777,
1026, 1031, 1045, 1050, 1055, 1061, 1067, 1073, 1081,
1107, 1147, 1194, 1865
.releaseEvents() 292, 298, 330, 360, 1144
.reload() 222, 735, 1167, 1865
.reload([true]) 1166
.remove() 456, 457, 742, 750, 762, 770, 771, 1198, 1865
.Remove() 1429, 1865
.RemoveAll() 1430, 1866
.removeAttribute() 238, 241, 259, 484, 564, 569, 575,
592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669,
684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728,
733, 738, 741, 747, 751, 755, 762, 768, 778, 827, 850,
863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1055,
1061, 1067, 1073, 1081, 1190, 1194, 1866
.removeAttributeNode() 238, 260, 484, 564, 569, 575,
592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669,
684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 729,

733, 738, 741, 747, 755, 762, 768, 778, 1027, 1031,
1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081,
1190, 1194, 1196, 1866
.removeBehavior() 240, 260, 484, 564, 569, 575, 592,
600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684,
689, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733,
738, 741, 747, 751, 755, 762, 768, 778, 1027, 1031,
1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081,
1190, 1194, 1196, 1867
.removeChild() 241, 260, 592, 600, 603, 612, 617, 621,
630, 661, 664, 669, 689, 694, 699, 702, 706, 711, 716,
720, 724, 729, 733, 741, 747, 762, 768, 778, 1027,
1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073,
1081, 1194, 1867
.removeExpression() 242, 261, 484, 564, 570, 575, 592,
600, 612, 617, 621, 630, 669, 684, 690, 694, 699, 702,
706, 711, 716, 720, 724, 729, 733, 747, 756, 762, 768,
778, 827, 850, 1027, 1031, 1035, 1039, 1045, 1050,
1055, 1061, 1067, 1073, 1081, 1194, 1867
.removeNamedItem() 267, 467, 1198, 1868
.removeNode() 240, 261, 592, 600, 603, 612, 617, 621,
630, 661, 664, 669, 690, 694, 699, 706, 711, 716, 720,
724, 729, 733, 741, 747, 756, 763, 768, 778, 1027,
1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073,
1081, 1194, 1868
.removeRule() 997, 1868
.repeat 885, 887, 893, 895, 898, 900, 904, 906, 909, 911,
915, 917, 918, 919, 922, 924, 927, 930, 932, 934, 946,
948, 949, 951, 954, 957, 966, 967, 971, 974, 1105,
1127, 1680, 1994
.repeatCount 885, 887, 890, 892, 893, 895, 896, 897,
898, 899, 900, 903, 904, 906, 907, 908, 909, 911, 913,
914, 915, 916, 917, 918, 919, 920, 921, 922, 924, 925,
926, 927, 928, 929, 930, 931, 932, 933, 934, 944, 945,
946, 948, 949, 951, 954, 956, 957, 964, 965, 966, 967,
969, 970, 971, 972, 973, 974, 1114, 1117, 1127, 1128,
1555, 1556, 1606, 1679, 1680, 1725, 1979, 1983, 1994,
1996
.repeatDur 872, 885, 892, 893, 896, 897, 898, 903, 904,
907, 908, 909, 913, 914, 915, 918, 920, 921, 922, 925,
926, 927, 928, 931, 932, 944, 945, 946, 949, 954, 964,
965, 969, 970, 971, 972, 1555, 1606, 1680, 1725
.replace() 218, 736, 1167, 1869
.replace(url) 1166
.replaceAdjacentText() 242, 261, 484, 564, 570, 575,
592, 601, 603, 612, 617, 621, 630, 651, 661, 664, 669,
684, 690, 694, 699, 702, 706, 711, 716, 720, 724, 729,
733, 738, 741, 747, 756, 763, 768, 778, 1027, 1031,
1035, 1039, 1045, 1050, 1055, 1061, 1068, 1073, 1081,
1190, 1194, 1869
.replaceChild() 241, 262, 593, 601, 603, 612, 617, 621,
630, 662, 664, 669, 690, 694, 699, 702, 706, 711, 716,
720, 724, 729, 733, 742, 747, 763, 769, 778, 1027,
1031, 1035, 1039, 1045, 1050, 1055, 1061, 1068, 1074,
1081, 1194, 1870
.replaceData() 280, 603, 1870
.replaceNode() 240, 262, 593, 601, 603, 612, 617, 621,
630, 662, 665, 669, 690, 694, 699, 706, 711, 716, 720,
724, 729, 733, 742, 747, 756, 763, 769, 778, 1027,
1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074,
1081, 1194, 1870
.Reset() 1379
.reset() 622, 630, 1870
.resetElement() 886, 887, 894, 895, 899, 900, 906, 910,
911, 916, 917, 919, 920, 923, 924, 929, 930, 933, 934,



947, 948, 949, 950, 951, 956, 957, 973, 974, 1124, 1127, 1871, 1991, 1994
 .resizeBy() 330, 399, 1872
 .resizeTo() 331, 399, 1872
 .restart 885, 893, 898, 904, 909, 915, 922, 927, 932, 946, 949, 954, 966, 971, 1681
 .resume() 593, 1873
 .resumeElement() 886, 894, 899, 906, 910, 916, 923, 929, 933, 947, 950, 956, 966, 973, 1873
 .ResumeSpeaking() 1515
 .returnValue 383, 406, 409, 1105, 1107, 1153, 1159, 1681, 1772
 .rev 561, 562, 572, 736, 1167, 1633, 1682
 .reverse() 167, 1874
 .right 462, 1087, 1682
 .rightContext 1187
 .rightMargin 590, 1682
 .RootFolder 1435, 1682
 .Rotation 514, 2004
 .round() 199, 1874
 .routeEvent() 300, 331, 360, 1145
 .RowDelim 1378
 .rowIndex 1059, 1682
 .rows 660, 1076, 1078, 1204, 1595, 1682, 1683
 .rowSpan 1065, 1071, 1683
 .rules 1017, 1205, 1683
 .save() 976, 1874
 .savePreferences() 1168
 .scheme 750, 1683
 .scrollIntoView() 736
 .scope 1065, 1071, 1683
 .scopeName 239, 246, 482, 562, 568, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1683
 .screen 383, 1684
 .screenLeft 383, 1684
 .screenX 1105, 1107, 1159, 1684
 .screenY 1105, 1107, 1159, 1160, 1684
 .ScriptEngine() 40, 1875
 .ScriptEngineBuildVersion() 40, 1875
 .ScriptEngineMajorVersion() 40, 1875
 .ScriptEngineMinorVersion() 40, 1876
 .scripts 1205
 .scroll 590, 610, 663, 1684
 .scroll() 400, 401, 1876
 .scrollAmount 744, 1684
 .scrollBy() 400, 1876
 .scrollDelay 744, 745, 1684, 1721
 .scrollHeight 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1684
 .scrolling 598, 649, 667, 750, 1684
 .scrollIntoView() 457, 460, 484, 564, 570, 575, 601, 603, 612, 617, 621, 631, 669, 684, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 771, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1085, 1194, 1201, 1876
 .scrollLeft 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 714, 718, 722, 726, 731, 744, 748, 750, 754, 760, 766, 775, 1018, 1029, 1033,

1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685, 1906, 1907
 .scrollTo() 400, 401, 1876
 .scrollTop 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 742, 745, 748, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685, 1906, 1907
 .scrollWidth 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 .search 562, 572, 735, 1166, 1167, 1685
 .search() 218, 1877
 .sectionRowIndex 1059, 1065, 1685
 .securityPolicy 1168
 .seekActiveTime() 886, 887, 895, 899, 906, 907, 910, 911, 913, 916, 917, 920, 921, 923, 924, 926, 929, 930, 933, 934, 944, 947, 949, 950, 956, 957, 966, 967, 970, 973, 974, 1129, 1590, 1877, 1996
 .seekSegmentTime() 886, 887, 895, 899, 906, 908, 910, 911, 913, 916, 917, 920, 921, 923, 924, 926, 929, 930, 933, 934, 944, 947, 949, 950, 956, 957, 967, 970, 973, 974, 1129, 1590, 1878, 1997
 .seekTo() 886, 887, 895, 899, 906, 908, 910, 911, 913, 916, 917, 919, 920, 921, 923, 924, 926, 929, 930, 933, 934, 944, 947, 949, 951, 956, 957, 967, 970, 973, 974, 1129, 1590, 1879, 1997
 .seekToFrame() 886, 887, 895, 899, 906, 908, 910, 911, 913, 916, 917, 919, 920, 921, 923, 924, 926, 929, 930, 933, 934, 944, 948, 949, 951, 956, 957, 967, 970, 973, 974, 1129, 1590, 1880, 1997
 .segmentDur 890, 1686
 .segmentTime 890, 1686
 .segmentTimeToActiveTime() 886, 895, 899, 906, 910, 916, 919, 923, 929, 933, 948, 951, 956, 967, 973, 1882
 .segmentTimeToSimpleTime() 887, 895, 899, 906, 910, 916, 919, 923, 929, 933, 948, 951, 956, 967, 973, 1882
 .select() 457, 460, 634, 635, 641, 690, 694, 699, 706, 711, 716, 720, 725, 729, 770, 771, 1082, 1085, 1883
 .selected 766, 1687
 .selectedIndex 760, 1687
 .selector 998, 1687
 .self 384, 649, 1687
 .SerialNumber 1435, 1687
 .setActive() 401, 446, 484, 564, 570, 575, 593, 601, 612, 618, 621, 631, 651, 662, 669, 684, 690, 694, 699, 702, 707, 711, 716, 720, 725, 729, 734, 747, 756, 763, 778, 937, 948, 951, 956, 1027, 1031, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1194, 1884, 1886
 .setAttribute() 238, 262, 484, 564, 570, 575, 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 .setAttributeNode() 238, 263, 484, 565, 570, 575, 593, 601, 603, 612, 618, 622, 631, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 751, 752, 756, 763, 769, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1190, 1194, 1196, 1887
 .SetBackgroundColor() 1516
 .setCapture() 484, 565, 570, 575, 593, 601, 612, 618, 622, 631, 684, 690, 695, 699, 702, 707, 711, 716, 720,



725, 729, 734, 736, 742, 747, 756, 763, 769, 778,
1027, 1031, 1045, 1051, 1056, 1062, 1068, 1074, 1082,
1107, 1147, 1194
.SetControlPosition() 1516
.setData() 416, 1104, 1112, 1115, 1887, 1981
.setDate() 183, 1888
.setEndPoint() 460, 1085, 1888
.setExpression() 242, 264, 484, 565, 570, 575, 593, 601,
612, 618, 622, 631, 669, 684, 690, 695, 699, 702, 707,
712, 716, 721, 725, 729, 734, 747, 756, 763, 769, 778,
828, 849, 850, 1027, 1031, 1035, 1039, 1045, 1051,
1056, 1062, 1068, 1074, 1082, 1195, 1888
.setFullYear() 183, 1892
.setHomePage() 857, 1892
.setHours() 183, 1892
.setInterval() 390, 401, 1747, 1892
.SetKey() 1516
.setMilliseconds() 184, 1894
.setMinutes() 184, 1895
.setMonth() 184, 1895
.SetMouthAnimation() 1516
.SetMouthColor() 1516
.setNamedItem() 268, 467, 1198, 1895
.SetOpaque() 1517
.setResizable() 440
.setSeconds() 184, 1895
.SetText() 1517
.SetTextAnimation() 1517
.SetTextColor() 1518
.SetTextPosition() 1518
.SetTextSize() 1518
.setTime() 185, 1895
.setTimeout() 391, 404, 1748, 1895
.setUTCDate() 185, 1898
.setUTCFullYear() 185, 1898
.setUTCHours() 185, 1898
.setUTCMilliseconds() 186, 1899
.setUTCMinutes() 186, 1899
.setUTCMonth() 186, 1899
.setUTCSeconds() 186, 1899
.setYear() 186, 1899
.setZOptions() 440
.shadowOpacity 515, 2004
.shape 560, 562, 571, 573, 1598, 1688
.ShareName 1435, 1688
.shift() 171, 173, 1855, 1899
.shiftKey 1105, 1107, 1153, 1154, 1688
.shiftLeft 1105, 1107, 1153, 1154, 1689
.ShortName 1438, 1440, 1689
.ShortPath 1438, 1440, 1689, 1690
.show() 393, 1178, 1182, 1757, 1899
.ShowBrowserUI() 1165, 1901
.showHelp() 405, 1901
.showModalDialog() 378, 379, 380, 381, 383, 405, 417,
1608, 1609, 1610, 1681, 1901
.showModelessDialog() 378, 380, 383, 408, 417, 1608,
1609, 1610, 1681, 1903
.siblingAbove 329
.siblingBelow 329
.simpleDur 890, 1690
.simpleTime 890, 1691
.simpleTimeToSegmentTime() 887, 895, 899, 906, 910,
916, 919, 923, 929, 933, 948, 951, 956, 967, 973, 1904
.sin() 199, 1905
.size 619, 687, 692, 697, 704, 709, 714, 718, 722, 726,
760, 1691, 1692

.Size 1438, 1441, 1692
.sizingMethod 515, 2005
.SizingMethod 512, 515, 2001, 2005
.Skip() 1444, 1905
.SkipLine() 1444, 1905
.slice() 167, 219, 1905
.slideStyle 515, 2005
.small() 219, 1906
.Sort 1378
.sort() 168, 1906
.source 1185
.sourceIndex 482, 562, 568, 573, 590, 598, 602, 610,
615, 619, 628, 649, 660, 663, 667, 682, 687, 692, 697,
700, 704, 709, 714, 718, 722, 726, 731, 736, 740, 745,
750, 754, 760, 775, 1000, 1018, 1029, 1033, 1037,
1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192,
1692
.sourceURL 860, 1692
.span 1033, 1037, 1692
.SpecialFolders() 1447
.specified 238, 246, 465, 1693
.speed 885, 890, 893, 898, 905, 909, 915, 922, 927, 928,
932, 946, 954, 966, 971, 972, 1693, 1725
.splice() 169, 1906
.split() 219, 1906
.spokes 515, 2005
.sqrt() 199, 1906
.SQRT1_2 199, 1694
.SQRT2 199, 1694
.squaresX 515, 2005
.squaresY 515, 2005
.src 327, 329, 568, 583, 649, 667, 671, 682, 704, 909,
915, 922, 927, 937, 946, 971, 1188, 1195, 1694, 1695
.SRC 516, 2006
.srcElement 1105, 1107, 1153, 1159, 1695, 1772
.srcElement.id 1103
.srcElement.name 1103
.srcFilter 1105
.standby 754, 1696
.start 682, 686, 704, 1256, 1696
.start() 743, 747, 1906
.StartColor 516, 2006
.StartColorStr 516, 2006
.startDownload() 856, 1906
.StartSpeaking() 1518
.StartSpeakingImmediate() 1519
.startX 516, 2006
.startY 516, 2006
.state 889, 890, 1696
.stateString 889, 890, 1698
.status 384, 516, 692, 714, 1078, 1699, 2006
.stop() 518, 743, 748, 859, 1675, 1854, 1855, 1907, 2008
.StopSpeaking() 1519
.strength 516, 2006
.stretchStyle 516, 2006
.strike() 219, 1907
.String() 148, 158, 214, 1907
.style 517, 735, 976, 1323, 2006
.style.accelerator 809, 818, 1918
.style.background 809, 818, 1918
.style.backgroundAttachment 809, 818, 1918
.style.backgroundColor 809, 818, 1918
.style.backgroundImage 809, 818, 1919
.style.backgroundPosition 809, 819, 1919
.style.backgroundPositionX 809, 819, 1919
.style.backgroundPositionY 809, 819, 1919



.style.backgroundRepeat 809, 819, 1919
 .style.behavior 809, 819, 1919
 .style.border 809, 819, 1920
 .style.borderBottom 809, 819, 1921
 .style.borderBottomColor 809, 819, 1921
 .style.borderBottomStyle 809, 819, 1921
 .style.borderBottomWidth 809, 819, 1922
 .style.borderCollapse 809, 819, 1922
 .style.borderColor 809, 819, 1922
 .style.borderLeft 809, 819, 1922
 .style.borderLeftColor 809, 819, 1923
 .style.borderLeftStyle 809, 819, 1923
 .style.borderLeftWidth 809, 819, 1923
 .style.borderRight 810, 819, 1924
 .style.borderRightColor 810, 819, 1924
 .style.borderRightStyle 810, 819, 1924
 .style.borderRightWidth 810, 819, 1925
 .style.borderStyle 810, 819, 1925
 .style.borderTop 810, 819, 1925
 .style.borderTopColor 810, 819, 1926
 .style.borderTopStyle 810, 819, 1926
 .style.borderTopWidth 810, 819, 1926
 .style.borderWidth 810, 819, 1927
 .style.bottom 810, 813, 817, 819, 823, 1927, 1940
 .style.clear 810, 811, 817, 818, 819, 820, 824, 1927, 1931, 1944
 .style.clientCaps Behavior 852
 .style.clip 810, 812, 816, 820, 822, 825, 1928, 1930, 1937, 1948
 .style.color 810, 820, 1928
 .style.cssText 810, 817, 820, 1929
 .style.cursor 810, 820, 1930
 .style.direction 810, 816, 818, 820, 825, 1930, 1947
 .style.display 797, 809, 810, 812, 816, 818, 820, 822, 825, 849, 1631, 1917, 1928, 1930, 1937, 1948
 .style.download Behavior 855
 .style.filter 810, 820, 1931
 .style.float 810, 817, 820, 824, 1927, 1944
 .style.font 811, 820, 1931
 .style.fontFamily 811, 820, 1932
 .style.fontSize 811, 820, 1932
 .style.fontStyle 811, 820, 1932
 .style.fontVariant 811, 820, 1932
 .style.fontWeight 811, 820, 1932
 .style.height 797, 811, 816, 820, 826, 849, 908, 914, 921, 927, 931, 945, 954, 971, 1631, 1655, 1932, 1949
 .style.height 811, 813, 817, 820, 823, 1932, 1940
 .style.homePage Behavior 856
 .style.httpFolder Behavior 857
 .style.imeMode 811, 820, 1933
 .style.layoutFlow 811, 816, 820, 826, 1933, 1949
 .style.layoutGrid 811, 820, 1933
 .style.layoutGridChar 811, 820, 1933
 .style.layoutGridLine 811, 821, 1933
 .style.layoutGridMode 811, 821, 1933
 .style.layoutGridType 811, 821, 1934
 .style.left 797, 811, 813, 814, 817, 821, 823, 824, 977, 1934, 1940, 1941
 .style.letterSpacing 811, 821, 1934
 .style.lineBreak 811, 821, 1934
 .style.lineHeight 811, 821, 1934
 .style.listStyle 811, 821, 1934
 .style.listStyleImage 811, 821, 1935
 .style.listStylePosition 811, 821, 1935
 .style.listStyleType 811, 821, 1935
 .style.listStyleTypeImage 811, 821, 1935

.style.margin 811, 821, 1935
 .style.marginBottom 811, 821, 1936
 .style.marginLeft 812, 821, 1936
 .style.marginRight 812, 821, 1936
 .style.marginTop 812, 821, 1936
 .style.mediaBar Behavior 858, 859, 860, 861, 862, 1119, 1124, 1126, 1130, 1568, 1603, 1613, 1620, 1621, 1631, 1659, 1660, 1666, 1675, 1692, 1778, 1793, 1854, 1855, 1907, 1986, 1991, 1993, 1998
 .style.minHeight 812, 822, 1937
 .style.onOffBehavior 812, 817, 822, 1937
 .style.overflow 810, 812, 815, 816, 820, 822, 825, 1928, 1930, 1937, 1946, 1948
 .style.overflowX 810, 812, 816, 820, 822, 825, 1928, 1930, 1937, 1948
 .style.overflowY 810, 812, 816, 820, 822, 825, 1928, 1930, 1937, 1948
 .style.padding 812, 822, 1938
 .style.paddingBottom 812, 822, 1938
 .style.paddingLeft 813, 823, 1938
 .style.paddingRight 813, 823, 1939
 .style.paddingTop 813, 823, 1939
 .style.pageBreakAfter 813, 816, 823, 826, 1939, 1940, 1948, 1949
 .style.pageBreakBefore 813, 816, 823, 826, 1939, 1948, 1949
 .style.pixelBottom 810, 813, 817, 819, 823, 1927, 1940
 .style.pixelHeight 813, 817, 823, 1940
 .style.pixelHeight 811, 813, 817, 820, 823, 1932, 1940
 .style.pixelLeft 811, 813, 814, 817, 821, 823, 824, 1934, 1940, 1941
 .style.pixelRight 813, 814, 817, 823, 824, 1940, 1941, 1942
 .style.pixelTop 813, 814, 816, 817, 823, 824, 825, 1940, 1941, 1947
 .style.pixelWidth 813, 814, 816, 817, 823, 824, 826, 1940, 1942, 1948
 .style.posBottom 810, 813, 817, 819, 823, 1927, 1940
 .style.posHeight 813, 817, 823, 1940
 .style.posHeight 811, 813, 814, 817, 820, 823, 824, 1932, 1940, 1941
 .style.position 797, 811, 814, 816, 817, 821, 824, 826, 849, 1631, 1934, 1941, 1949, 1950
 .style.posLeft 811, 813, 814, 817, 823, 824, 1934, 1940, 1941
 .style.posRight 813, 814, 817, 823, 824, 1940, 1941, 1942
 .style.posTop 813, 814, 816, 817, 823, 824, 825, 1940, 1941, 1947
 .style.posWidth 813, 814, 816, 817, 823, 824, 826, 1940, 1941, 1942, 1949
 .style.right 813, 814, 817, 823, 824, 1940, 1941, 1942
 .style.rubyAlign 814, 824, 1942
 .style.rubyOverhang 814, 824, 1942
 .style.rubyPosition 814, 824, 1942
 .style.saveFavorite Behavior 862
 .style.saveHistory Behavior 863
 .style.scrollbar3dLightColor 814, 824, 1942
 .style.scrollbarArrowColor 814, 824, 1942
 .style.scrollbarBaseColor 814, 824, 1942
 .style.scrollbarDarkShadowColor 814, 824, 1943
 .style.scrollbarFaceColor 814, 824, 1943
 .style.scrollbarHighlightColor 815, 824, 1943
 .style.scrollbarShadowColor 815, 824, 1944
 .style.scrollbarTrackColor 815, 824, 1944



.style.styleFloat 810, 811, 815, 817, 818, 820, 824, 1927, 1931, 1944
 .style.tableLayout 812, 815, 822, 824, 1937, 1944
 .style.textAlign 815, 824, 1944, 1945
 .style.textAlignLast 815, 824, 1944, 1945
 .style.textAutospace 815, 824, 1945
 .style.textDecoration 815, 817, 824, 1945
 .style.textDecorationBlink 815, 817, 824, 1945
 .style.textDecorationLineThrough 815, 817, 824, 1945
 .style.textDecorationNone 815, 817, 825, 1945
 .style.textDecorationOverline 815, 818, 825, 1945
 .style.textDecorationUnderline 815, 818, 825, 1945
 .style.textIndent 815, 825, 1946
 .style.textJustify 815, 824, 825, 1944, 1945, 1946
 .style.textOverflow 812, 815, 822, 825, 1937, 1946
 .style.textTransform 815, 825, 1946
 .style.textUnderlinePosition 815, 825, 1947
 .style.time2 Behavior 560, 561, 562, 563, 571, 572, 573, 575, 597, 598, 599, 609, 610, 614, 615, 618, 619, 627, 628, 666, 667, 681, 682, 686, 687, 688, 691, 692, 695, 696, 697, 700, 703, 704, 708, 709, 712, 713, 714, 717, 718, 721, 722, 725, 726, 727, 743, 744, 745, 765, 767, 774, 775, 776, 865, 890, 1076, 1077, 1078, 1107, 1114, 1117, 1120, 1121, 1124, 1125, 1127, 1128, 1129, 1131, 1191, 1192, 1468, 1555, 1557, 1559, 1560, 1562, 1568, 1573, 1579, 1583, 1584, 1586, 1588, 1590, 1593, 1603, 1606, 1616, 1617, 1619, 1621, 1623, 1626, 1628, 1629, 1630, 1631, 1632, 1641, 1642, 1646, 1647, 1648, 1649, 1655, 1656, 1657, 1658, 1666, 1669, 1670, 1673, 1675, 1676, 1680, 1681, 1686, 1690, 1691, 1693, 1698, 1703, 1705, 1706, 1707, 1709, 1713, 1718, 1719, 1723, 1725, 1726, 1728, 1729, 1730, 1732, 1734, 1736, 1743, 1760, 1764, 1765, 1845, 1847, 1850, 1851, 1853, 1855, 1857, 1871, 1873, 1877, 1878, 1879, 1880, 1882, 1904, 1979, 1984, 1987, 1988, 1991, 1992, 1994, 1995, 1996, 1997, 1999, 2000
 .style.time2 Behavior Import 876
 .style.time2 Behavior Medien zur Animation 874
 .style.time2 Behavior Time-Formate 872
 .style.time2 Behavior XML-Namensraum 875
 .style.time2 Objekt 1147
 .style.time2.animate Behavior-Objekt 890
 .style.time2.animateColor Behavior-Objekt 896
 .style.time2.animateMotion Behavior-Objekt 900
 .style.time2.animation Behavior-Objekt 907
 .style.time2.audio Behavior-Objekt 911
 .style.time2.excl Behavior-Objekt 917, 940
 .style.time2.img Behavior-Objekt 920
 .style.time2.media Behavior-Objekt 924
 .style.time2.playItem Behavior-Objekt 934, 937
 .style.time2.playlist Behavior-Collection 934, 937, 939, 948, 951, 956, 1561, 1884
 .style.time2.priorityClass Behavior-Objekt 940
 .style.time2.ref Behavior-Objekt 944
 .style.time2.set Behavior-Objekt 949
 .style.time2.switch Behavior-Objekt 957
 .style.time2.transitionFilter Behavior-Objekt 508, 959, 965, 966, 1586, 1588, 1619, 1629, 1657, 1703, 1719, 1723, 1730
 .style.time2.video Behavior-Objekt 967
 .style.top 813, 814, 815, 816, 817, 823, 824, 825, 1940, 1941, 1947
 .style.unicodeBidi 810, 816, 818, 820, 825, 1930, 1947
 .style.userData Behavior 974
 .style.verticalAlign 816, 825, 826, 1947, 1949

.style.visibility 509, 528, 672, 810, 812, 816, 820, 822, 825, 1928, 1930, 1937, 1948
 .style.whiteSpace 810, 811, 813, 815, 816, 817, 818, 820, 823, 824, 825, 826, 1927, 1931, 1939, 1944, 1946, 1948, 1949
 .style.width 797, 810, 813, 814, 816, 817, 818, 820, 823, 824, 826, 849, 908, 914, 921, 927, 931, 945, 954, 971, 1631, 1655, 1931, 1940, 1942, 1944, 1948, 1949
 .style.wordBreak 813, 816, 823, 826, 1939, 1948, 1949
 .style.wordSpacing 816, 826, 1949
 .style.wordWrap 813, 816, 823, 826, 1939, 1940, 1948, 1949
 .style.writingMode 797, 811, 816, 820, 826, 849, 1631, 1933, 1949
 .style.zIndex 816, 826, 1950
 .style.zoom 797, 816, 826, 849, 1950, 1951
 .style.pixelBottom 813, 817, 823, 1940
 .styleSheets 1205
 .sub() 219, 1907
 .SubFolders 1441, 1703
 .submit() 622, 631, 1908
 .substr() 219, 1908
 .substring() 219, 1908
 .substringData() 603, 1909
 .subtype 966, 1703
 .suffixes 1169, 1171
 .summary 1018, 1704
 .sup() 219, 1909
 .swapNode() 240, 266, 565, 570, 575, 593, 601, 603, 613, 618, 622, 632, 651, 662, 665, 670, 684, 690, 695, 699, 702, 707, 712, 716, 721, 725, 729, 734, 738, 742, 748, 756, 763, 769, 778, 1027, 1032, 1035, 1039, 1045, 1051, 1056, 1062, 1068, 1074, 1082, 1190, 1195, 1909
 .syncBehavior 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 745, 767, 775, 885, 909, 911, 915, 917, 918, 922, 924, 927, 930, 932, 934, 946, 949, 954, 957, 971, 972, 974, 1078, 1131, 1192, 1705, 1706, 1707, 1999
 .syncmaster 885, 909, 915, 918, 922, 927, 932, 946, 954, 971, 1705
 .syncMaster 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 745, 766, 775, 909, 915, 918, 922, 927, 932, 946, 954, 971, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1706
 .syncTolerance 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 872, 909, 915, 918, 922, 927, 932, 946, 954, 971, 1078, 1192, 1706, 1707
 .systemBitrate 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 957, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1707
 .systemCaptions 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 957, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1707
 .systemLanguage 412, 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 854, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 957, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1170, 1192, 1708



.systemOverdubOrSubtitle 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 957, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708
 .tabIndex 482, 562, 568, 573, 590, 598, 610, 615, 619, 628, 649, 660, 667, 682, 687, 692, 697, 704, 709, 714, 718, 722, 727, 731, 745, 754, 760, 776, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708
 .tagName 240, 247, 482, 562, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 665, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 735, 736, 740, 745, 750, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1708
 .tags() 136, 274, 275, 448, 450, 451, 452, 453, 455, 456, 461, 473, 474, 566, 571, 614, 641, 642, 685, 707, 738, 742, 757, 770, 890, 1040, 1046, 1085, 1171, 1190, 1197, 1909
 .tagUrn 239, 247, 482, 563, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1709
 .taintEnabled() 412, 1169, 1170, 1909
 .tan() 199, 1909
 .target 563, 573, 628, 735, 736, 1106, 1156, 1160, 1709
 .targetElement 894, 898, 905, 950, 966, 1709
 .tBodies 1206
 .test() 1185
 .text 459, 590, 602, 665, 735, 767, 1083, 1188, 1711
 .textKashidaSpace 815, 825, 1946
 .TextQualifier 1379
 .TextRange 1206
 .TextRectangle 1206
 .tFoot 1018, 1711
 .tHead 1018, 1711
 .timeAction 610, 885, 909, 915, 922, 927, 932, 946, 954, 972, 1711
 .timeAll 1713
 .timeChildren Collection 451, 593
 .timeContainer 563, 573, 598, 610, 615, 619, 628, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 955, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1713
 .timeParent 885, 894, 898, 905, 909, 915, 922, 927, 932, 946, 950, 955, 972, 1713
 .timeStartRule 590, 1714
 .title 428, 482, 563, 568, 573, 590, 599, 610, 615, 628, 649, 660, 667, 682, 688, 692, 697, 704, 709, 714, 718, 722, 727, 731, 740, 745, 754, 776, 909, 915, 922, 927, 937, 946, 972, 996, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1714, 1715, 1716, 1718
 .to 893, 894, 897, 898, 900, 904, 905, 950, 966, 1584, 1718, 1719
 .toArray() 1428, 1429, 1834, 1835
 .toDateString() 186, 1909
 .toElement 1106, 1107, 1159, 1720
 .toExponential() 205, 1909
 .toFixed() 205, 1909
 .toGMTString() 187, 1910
 .toLocaleDateString() 187, 1910
 .toLocaleLowerCase 220, 1910
 .toLocaleString() 158, 187, 212, 1910

.toLocaleTimeString() 187, 1910
 .toLocaleUpperCase() 220, 1911
 .toLowerCase() 220, 1911
 .top 329, 388, 462, 1087, 1720, 1721
 .topMargin 590, 1721
 .toPrecision() 205, 1911
 .toString() 148, 159, 170, 175, 208, 212, 214, 1911
 .TotalSize 1435, 1721
 .toTimeString() 187, 1911
 .toUpperCase() 220, 1912
 .toUTCString() 187, 975, 1912
 .transition 517, 2007
 .trueSpeed 744, 745, 1684, 1721
 .type 458, 563, 599, 633, 634, 635, 636, 637, 638, 641, 736, 754, 761, 770, 798, 826, 909, 915, 922, 928, 937, 946, 966, 972, 996, 1078, 1083, 1106, 1107, 1153, 1156, 1159, 1160, 1169, 1171, 1188, 1721, 1722, 1723, 1724, 1757, 1772
 .Type 1438, 1441, 1724
 .typeDetail 770, 1724
 .unescape 159, 213, 1912
 .unescape() 149, 152, 1758
 .uniqueID 239, 247, 430, 481, 561, 563, 567, 568, 572, 573, 590, 597, 599, 601, 602, 615, 619, 627, 628, 648, 649, 659, 660, 662, 663, 666, 667, 681, 682, 687, 688, 691, 692, 696, 697, 700, 701, 703, 704, 708, 709, 713, 714, 717, 718, 721, 722, 726, 727, 730, 731, 736, 737, 739, 740, 744, 745, 750, 751, 753, 754, 761, 767, 776, 1016, 1018, 1028, 1029, 1032, 1033, 1036, 1037, 1041, 1042, 1047, 1048, 1052, 1053, 1058, 1059, 1064, 1065, 1070, 1071, 1077, 1078, 1188, 1191, 1192, 1195, 1635, 1724
 .unshift() 171, 173, 1912
 .unterklasse_name{eigenschaften_liste} 978
 .updateInterval 1182, 1183, 1725
 .updateMode 909, 915, 922, 928, 946, 972, 1725
 .URL 430, 917, 928, 930, 972, 1107, 1131, 1726, 2000
 .URLUnencoded 430, 1726
 .urn 507, 563, 1726
 .urns() 271, 274, 448, 450, 451, 452, 453, 455, 456, 471, 474, 566, 571, 614, 632, 641, 642, 685, 707, 738, 742, 757, 763, 770, 1000, 1040, 1046, 1190, 1197, 1913
 .useMap 682, 686, 754, 1726
 .userAgent 412, 1168, 1170, 1726
 .userLanguage 412, 854, 1170, 1726, 1727
 .userProfile 1170, 1727
 .UTC() 187, 1913
 .vAlign 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1727
 .value 238, 247, 466, 599, 633, 634, 635, 637, 638, 641, 686, 688, 692, 697, 701, 705, 709, 714, 718, 723, 727, 761, 767, 1078, 1721, 1727
 .valueOf() 149, 159, 208, 213, 214, 1913
 .values 893, 894, 897, 898, 900, 904, 905, 965, 966, 1562, 1584, 1628, 1670, 1728, 1729, 1730
 .vcard_name 709, 727, 1731
 .version 663, 1731
 .visibilitiy 327
 .visibility 329, 509, 528, 672
 .vLink 590, 1731
 .vlinkColor 430, 1731
 .volume 583, 885, 890, 909, 915, 922, 928, 932, 946, 955, 972, 1731, 1732
 .VolumeName 1435, 1733
 .vspace 568, 667, 671, 682, 686, 745, 754, 1733
 .wheelDelta 1107, 1733



.which 1100, 1106, 1156, 1160
 .width 329, 649, 660, 667, 671, 682, 688, 693, 697, 705,
 709, 714, 718, 723, 727, 745, 754, 803, 854, 1018,
 1033, 1037, 1060, 1066, 1072, 1106, 1160, 1182, 1183,
 1733
 .wipeStyle 517, 2007
 .WipeStyle 517, 2007
 .wmf 703
 .wrap 1076, 1078, 1595, 1733
 .write() 437, 447, 1849, 1913
 .Write() 1445, 1913
 .WriteBlankLines() 1445, 1914
 .WriteLine() 1445, 1914
 .writeln() 437, 447, 1849, 1914
 .writingMode 508
 .x 735, 1106, 1159, 1160
 .xbm 703
 .XMLDocument 239, 249, 430, 863, 864, 1188, 1195,
 1734
 .Xray 517, 2007
 .XSLDocument 238, 249, 430, 1734
 .y 735, 1106, 1159, 1160
 .z-index 757
 .zIndex 329
 .zoom 1951
 / 75, 1101
 /* */ 977
 /* */ 53
 // 53
 /= 76
 : 75
 :active 809, 818, 1917
 :first 818, 983, 998, 1553, 1678, 1917
 :first-letter 809, 818, 1917
 :first-line 809, 818, 1917
 :footer 983, 998, 1553, 1678
 :header 983, 998, 1553, 1678
 :hover 809, 818, 1917
 :left 818, 983, 998, 1553, 1678, 1917
 :left : footer 983, 998, 1553, 1678
 :left : header 983, 998, 1553, 1678
 :link 809, 818, 1917
 :right 818, 1917
 :right 983
 :right 998
 :right 1553
 :right 1678
 :right:footer 983, 998, 1553, 1678
 :right:header 983, 998, 1553, 1678
 :visited 809, 818, 1917
 --; 76
 ? 75, 562, 573, 1101, 1167, 1685
 Operator 66, 80
 ?: Operator 91
 ?IMPORT 506, 507, 869, 876, 877
 ?searchtext 1166
 @ 1101
 @_alpha 100
 @_jscript 100
 @_jscript_build 100
 @_jscript_version 100
 @_mac 100
 @_mc680x0 100
 @_PowerPC 100
 @_win16 100
 @_win32 100

@_x86 100
 @cc_on 99
 @eigenschaften 977
 @elif 100
 @else 100
 @end 99, 100
 @if 100
 @import 982
 @import url 997
 @import url() 996, 1738
 @media 982
 @media all 479
 @page 818, 996, 998, 1553, 1738, 1917
 @page Pseudoklasse 998
 @page Regel 998, 1678
 @set 101
 @varname 101
 [1101
 [] 75
 [^0] 1184
 [^9] 1184
 [^A-Za-z0-9]zeichenfolge 1184
 [^\f\n\r\t\v]zeichenfolge 1184
 [^zeichen_menge] 1184
 [A-Za-z0-9]zeichenfolge 1184
 [\f\n\r\t\v]zeichenfolge 1184
 [zeichen_menge] 1184
] 1101
 ^ 75, 76, 1101
 ^zeichenfolge 1183
 _ 1101
 _blank 396, 437, 735, 1709
 _media 396, 437, 858, 1709
 _parent 396, 438, 735, 1164, 1709
 _search 396, 438, 735, 1709
 _self 396, 438, 735, 1164, 1709
 _top 221, 396, 438, 735, 1164
 ` 1101
 { 1101
 { } 85
 | 75, 76, 1101
 || 75
 } 1101
 ~ 76, 1101
 “de“ 748
 “en“ 748
 + 75, 76, 214, 1101
 + - / * 75
 ++ 75
 ++; 76
 += 75, 76
 < 75, 1101
 < > 76
 <!-- //--> 977
 <![endif]--> 38, 39
 <!--[if ! IE 5]> 39
 <!--[if IE 5]> 38, 39
 <!DOCTYPE HTML PUBLIC ...> 802
 306, 449, 565
 </APPLET> 306, 450, 567, 570
 </BODY> 584
 </EMBED> 309, 452, 613, 757
 </FORM> 622
 </MAP> 571, 1599
 <?IMPORT 869, 876, 877
 << 75, 76



<< Operator 79
 <<< 76
 <= 75, 76
 <A> 306, 448, 565
 <A> 561, 562, 572, 736, 984, 1098, 1167, 1633, 1679
 <APPLET> 306, 450, 566, 570
 <AREA> 571, 1598
 <BG SOUND> 576
 <BODY> 584
 <BODY> 1097, 1098
 <DIV> 1098, 1099
 <EMBED> 309, 451, 581, 613, 756
 <FORM> 622
 <FORM> 1104
 <FRAME> 1097
 <FRAMESET> 1097, 1104
 <H1> bis <H6> 984
 <HEAD> 70
 <HTML XMLNS:> 480
 <ILAYER> 327
 324, 670
 1097, 1098, 1099
 <INPUT> 325, 670
 <INPUT> 1097, 1098
 <LAYER> 326
 <MAP> 571, 1598
 <META> 748
 <META HTTP-EQUIV> 748, 1596, 1635
 <META> 509
 <NOSCRIPT> 41
 <OBJECT> 43, 44, 71
 <OPTION> 1098
 <P> 984
 <PARAM> 306, 309, 450, 452, 566, 570, 613, 756
 <SCRIPT FOR> 41, 1095
 <SCRIPT FOR= Event= 41, 1095
 <SCRIPT DEFER> 240
 <SCRIPT LANGUAGE> 46
 <SCRIPT> 40
 <SELECT> 1098
 1098, 1099
 <STYLE TYPE="text/css"> 977
 <STYLE> 796, 976
 <TEXTAREA> 1097
 <TITLE> 749
 <VAR> 1190
 = 76, 1101
 -= 76
 == 75, 76
 === Operator 78
 > 75
 >= 75, 76
 >> 75, 76
 >> Operator 80
 >>> 75, 76
 >>> Operator 80
 0 bis 9 1101
 01.01.1970 0 Uhr Weltzeit 176
 1.1.1970 0 Uhr 159, 213, 1913
 24-Stunden Digitaluhr 518
 2D Textzeichen-Layout-Gitter 811, 821, 1933
 2-dimensionalen Feld 163
 2D-Position 277, 476
 3.0B2 (Win16;I) 1167
 32-Bit-Basis 76
 32-Bit-Zahl 149

3D-Effekt 983, 987
 3D-Effekt eines Bildes 542
 3D-Element 981
 3D-Rahmen 991
 3D-Rahmen Farbe 1014, 1058, 1063, 1069, 1581
 3D-Schatten von Button im Dialogfenster 981
 4.1 (WinNT;I) 1167
 68K 411, 1601
 8.3-Kodierung Dateiname 48
 A 1108, 1224
 a bis z 1101
 a Objekt 448, 559, 565, 734
 a Objekt Events 1132, 1953
 a Objekt Filter 548
 a Objekt Pseudoklasse des Link, der bereits aktiviert wurde 809, 818, 1917
 a Objekt Pseudoklasse des Link, der nicht kürzlich aktiviert wurde 809, 818, 1917
 a Objekt Pseudoklasse für aktiven Link 809, 818, 1917
 a Objekt Pseudoklasse für Link, der nicht aktiviert wurde 809, 818, 1917
 a Objekt Styles 828
 Abarbeitung von Scripten 129, 1100, 1147
 Abbruch des Ladens 1095
 Abbruch Ladevorgang 1161
 Abfangen der Resetaktion 1103, 1104
 abfangen von Laufzeitfehlern 128
 Abfangen von Runtime Fehlern 128
 Abfolge von Zeitpunkten 865
 Abhak-Kästchen 633
 ableiten Objektvariable von Objektklasse 69
 Abmahnungsgefahr 319, 643, 653
 Abort 1153, 1156, 1159, 1160
 about:blank 396
 above 1016, 1627
 ABOVE 326
 Absatz 984, 985
 Abschaltung der automatischen Umrandung von angeklickten Objekten IE 1775
 Abschaltung von Cookies 486
 Abschießen des Formulars 638
 abschießen eines Formulars 1104
 Absolutbetrag 199, 1734
 absolute 984
 absolute Pfade 24
 absolute Positionierung 277, 476
 AbsolutePosition 277, 476
 Abstand des Aussenrandes zur Umgebung 589, 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Abstand des Objekthaltens zum Aussenrand 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Abstand linker Objektrand zur Umgebung rechte Kante 811, 813, 814, 817, 821, 823, 824, 1934, 1940
 Abstand oberer Objektrand zur Umgebung Unterkante 813, 814, 815, 817, 823, 824, 825, 1940, 1941
 Abstand Objekt zum linken Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Abstand Objekt zum oberen Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759,



765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Abstand rechter Objektrand zur Umgebung linke Kante 813, 814, 817, 823, 824, 1940, 1941, 1942
 Abstand Textzeichen 811, 821, 1934
 Abstand unterer Objektrand zur Umgebung Oberkante 810, 813, 817, 819, 823, 1927, 1940
 Abstand von HTML-Elementen 984
 Abstand von linken sichtbaren Randes des Umgebungsobjektes Body 590, 615, 760, 766
 Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Body 590, 615, 760, 766
 Abstand zum Elternobjekt 666, 681, 1634
 Abstand zweier Objekte oder zweier Textzeilen 811, 821, 1934
 Abstand zwischen Objekt und Margin bzw. Rahmen links 813, 823, 1938
 Abstand zwischen Objekt und Margin bzw. Rahmen links, rechts, oben, unten 812, 822, 1938
 Abstand zwischen Objekt und Margin bzw. Rahmen oben 813, 823, 1939
 Abstand zwischen Objekt und Margin bzw. Rahmen rechts 813, 823, 1939
 Abstand zwischen Objekt und Margin bzw. Rahmen unten 812, 1938
 Abstand zwischen Worten 816, 826, 1949
 Abstandsbereich 811, 821, 1935
 abstract 860, 861, 1778
 Abwärtskompatibilität der Scriptmaschine 18, 31, 42, 135
 ACCESSKEY 729
 Acrobat-Reader-Plugin 368, 1168, 1171
 ACTION 622, 695
 Active Setup 854
 ActiveX 1145
 Active-X 410, 854
 ActiveX Control Klassenbezeichner 752, 1591
 ActiveX Data Objects 1108
 ActiveX prüfen 1375
 ActiveX-Control 752, 908, 909, 914, 922, 927, 945, 971, 1376, 1672, 1674
 ActiveX-Control anstelle Plugin beim IE 451, 613, 756, 1171
 Active-X-Control des Mediaplayers 1287
 ActiveX-Control Direct Animation 1382
 ActiveX-Control TDC 1001
 ActiveX-Controls 1375
 Active-X-Komponenten des IE 854
 ActiveXObject Objekt 1420
 ADO 1108
 Adobe Acrobat-Plugin 369, 1169, 1170
 Adobe Acrobat-Reader-Plugin 368, 1168, 1171
 Adresse von Videoclip 681, 686, 703, 1620
 Adress-Eingabezeile 397, 439
 Adresszeile 397, 428, 439, 1679
 Advanced Stream Redirector (ASX)-Datei 934
 Advanced Stream Redirector Datei 860, 861, 907, 913, 917, 920, 925, 926, 930, 936, 940, 944, 949, 969, 970, 974, 1131, 1553, 1575, 1576, 1578, 1778, 1999
 Advanced Streaming Format (ASF)-Datei 917, 928, 930, 972, 1107, 1131, 1726, 2000
 Advanced Streaming Format Datei 917, 930, 1131, 2000
 Aktion des HTML-Elementes 1096
 Aktion des Objektes in der Timeline 610, 885, 909, 915, 922, 927, 932, 946, 954, 972, 1711

Aktion während der Timeline 867
 aktiv in der Timeline 1642
 aktiv setzen eines Fensters 401, 1886
 aktive Fenstertitelzeile 981
 aktive Selektion im Dokument 770
 aktiver Link Farbe 424, 1564
 aktives Fenster 981
 aktives Fensters 401, 1886
 aktivieren 1096
 aktivieren Eventdurchreichung 446, 484, 564, 570, 575, 593, 601, 612, 618, 621, 631, 651, 662, 669, 684, 690, 694, 699, 702, 707, 711, 716, 720, 725, 729, 734, 747, 756, 763, 778, 1027, 1031, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1194, 1884
 Aktivierung eines Objektes 1112, 1977
 Aktivität in der Timeline 889
 aktuelle Anzahl Farbbits pro Bildpunkt 1182
 aktuelle Bildschirmauflösung 1182
 aktuelle Breite 1182
 aktuelle Fenster 384, 1687
 aktuelle Höhe 1182
 aktuelle JScript-Maschine 40
 aktuelle Objekt-Instanz 77
 aktuelle Seite 1165
 aktuelle Seite mit neuer geladener Seite überschreiben 736, 1166
 aktueller Frame 319
 aktueller Frame Referenz 649, 1687
 aktueller Punkt auf Timeline 1691
 aktueller Status Body 590, 615, 760, 766
 aktueller Text der Statuszeile 384, 1699
 aktueller Zeitpunkt in der Timeline 889, 1560
 aktueller Zeitpunkt in der Timeline der Wiederholungen laut autoReverse 1686
 aktuelles Dokument 1165
 aktuelles Dokument neu laden 1167, 1865
 aktuelles Fenster 282, 395, 1849
 aktuelles Fenster Referenz 649, 1687
 Album 874, 1469
 alert-Box 1637
 Alert-Box 1637
 ALIGN 306, 325, 450, 566, 570, 670, 1938
 ALink 589
 Alink Farbe 1564
 Alink im Dokument 1564
 all 1017, 1620, 1655, 1683
 all.unterklasse_name{eigenschaften_liste} 978
 Alle Anker eines Dokumentes anzeigen 449, 565
 Alle Daten auf dem Server dürfen gescannt werden 50
 Alle Daten auf dem Server dürfen NICHT gescannt werden 50
 ALLE Suchmaschinen dürfen ALLES außer /temp/ scannen 51
 ALLE Suchmaschinen dürfen ALLES außer /test/test.htm scannen 51
 ALLE Suchmaschinen dürfen ALLES außer /test/test.htm UND /temp/ scannen 51
 ALLE Suchmaschinen dürfen ALLES scannen 50
 ALLE Suchmaschinen dürfen NICHTS scannen 50
 ALLE Suchmaschinen dürfen scannen 49
 Alle Textmarken (Anker) eines Dokumentes anzeigen 449, 565
 ALLES außer /temp/ scannen 51
 ALLES außer /test/ scannen 50
 ALLES außer /test/test.htm scannen 50, 51
 ALLES außer /test/test.htm UND /temp/ scannen 51



ALLES scannen 50
 allokieren 93
 ALLOWTRANSPARENCY 665
 Alpha 411, 1601
 Alpha Filter 518
 AlphaImageLoader Filter 520
 als bezüglich Anzeigefenster-Rand 984
 Alt 1105
 ALT 306, 325, 450, 566, 570, 670
 Alt + Taste 481, 560, 567, 571, 589, 597, 609, 614, 618, 680, 686, 691, 695, 703, 707, 712, 717, 721, 725, 730, 743, 752, 759, 774, 1014, 1027, 1032, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1190, 1556
 ALT_MASK 358, 363, 364, 1105, 1156, 1157
 ALT+F4 397, 439
 alternate 1581
 alternativer Text als Tooltip 571, 680, 686, 703, 1564
 ALT-Taste 1104, 1153
 ALT-Taste links 1104
 ALT-Taste links Status 1106, 1565
 ALT-Tasten-Status 1106, 1565
 alwaysLowered 438
 alwaysRaised 438
 an Url angehängte Textdaten 317, 323, 646, 657
 anchors[] 449, 566
 anchors[].name 449, 566
 ändern Fenstergröße 1103
 ändern Textbereich-Zeichen-Zeiger 460, 1084, 1085, 1839, 1842, 1888
 angeklicktes Button 637
 anzeigen transparent 811, 821, 1935
 anhängen Kind 250, 591, 599, 602, 610, 616, 619, 628, 660, 663, 667, 688, 693, 697, 705, 709, 714, 718, 723, 727, 732, 740, 745, 767, 776, 1019, 1029, 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1192, 1739
 anhängen Knoten 250, 591, 599, 602, 610, 616, 619, 628, 660, 663, 667, 688, 693, 697, 705, 709, 714, 718, 723, 727, 732, 740, 745, 767, 776, 1019, 1029, 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1192, 1739
 ANIMATE 890
 ANIMATECOLOR 896
 animatecolor Objekt 893, 897, 898, 904, 965, 1562, 1729
 ANIMATEMOTION 900
 animatemotion Objekt 904, 1666, 1670
 Animation 865, 887, 1101
 Animation eines beliebigen Elementes 907
 Animation eines Objektes 508
 Animation geladene Bilder 325, 670
 Animationen 1182
 animiertes Bild 325, 670, 1101
 Animierung des Filters 518, 2008
 Anker 241, 244, 333, 448, 559, 561, 565, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 734, 736, 738, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1166, 1167, 1188, 1191, 1631, 1633, 1661
 Anker anpringen 306, 449, 565
 Anker eines Dokumentes anzeigen 449, 565
 Anklicken des Restbuttons 637
 Anordnung im Dokument 985
 Anspringen eines Ankers 306, 333, 449, 565, 734, 1166
 Anweisung Blockanweisung 85
 Anweisung break 85, 92, 96

Anweisung continue 86, 92
 Anweisung delete 65, 143
 Anweisung do while 87
 Anweisung for 87
 Anweisung for .. in 174
 Anweisung for in 88, 150, 209, 1676
 Anweisung function 88, 95, 102, 105, 106, 161, 1553, 1567, 1589, 1650, 1739, 1745
 Anweisung if else 91
 Anweisung label 85, 86, 92
 Anweisung Leeranweisung 85
 Anweisung new 69, 111, 149, 160, 208
 Anweisung new A 72
 Anweisung return 95, 105, 106
 Anweisung switch 85, 96
 Anweisung this 84, 97, 150, 209
 Anweisung throw 97, 98, 129
 Anweisung try catch finally 97, 128
 Anweisung var 63, 67, 69, 99
 Anweisung while 86, 99
 Anweisung with 99, 150, 209
 Anweisungen 52, 84
 Anweisungen in Javascript 1.5 101
 Anweisungen in Javascript und JScript 85
 Anweisungen in JScript 85
 Anweisungen nur in JScript 99
 Anwendbarkeit Filter 512, 2001
 Anwendungsbereich des Cookie 488
 Anzahl aller Frames 382, 1651
 Anzahl aller IFrames 382, 1651
 Anzahl der Argumente 162, 1650
 Anzahl der Argumente einer Funktion 195, 1650
 Anzahl der Bits pro Pixel für eine Farbe 854, 1583
 Anzahl der Elemente in einem Feld 166, 1650
 Anzahl der Feldelemente also Feldlänge 451, 452, 453, 454, 455, 456, 457, 461, 507, 511, 593, 641, 642, 652, 685, 707, 738, 742, 759, 769, 771, 890, 940, 998, 999, 1000, 1040, 1046, 1085, 1108, 1165, 1171, 1190, 1650
 Anzahl der History-Einträge 1165
 Anzahl der horizontalen Bildpunkte 1183, 1607, 1652
 Anzahl der Millisekunden seit dem 1.1.1970 0 Uhr 159, 213, 1913
 Anzahl der Parameter 104, 160
 Anzahl der Parameter einer Funktion 195, 1650
 Anzahl der Plugins 368, 1168
 Anzahl der Scrollaktionen marquee Objekt 744
 Anzahl der verfügbaren Farben 1182
 Anzahl der vertikalen Bildpunkte 1183, 1608, 1652
 Anzahl der Zeichen im Kommentar 601, 1650
 Anzahl der Zeilen in einer List-Box 760, 1692
 Anzahl Radiobutton innerhalb der Gruppe 636
 Anzeige des Dokumentes beenden 1749
 Anzeige des Filters zurücksetzen 518, 2008
 Anzeige des Objektes mit Scrollelementen 812, 822, 1937
 Anzeigebereich 397, 439, 440
 Anzeigebereich im Browserfenster 1104
 Anzeigefenster-Rand 984
 Anzeigegeschwindigkeit 1182
 anzeigen Fenster 395, 437, 1849
 anzeigen horizontal 810, 820, 1930
 anzeigen HTTP-Verzeichnis 857
 Anzeigesprache 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 775, 1017,



1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Apache-HTTP-Server 24
 appCodeName 1167
 Apple QuickTime Movie 671, 702
 Applet 450, 570, 1101
 applet Objekt 566
 applet Objekt Events 1132, 1953
 applet Objekt Filter 548
 applet Objekt Styles 829
 Applet Url der *.class-Datei 567, 753, 1594
 Applet Url der Komponente 567, 753, 1594
 Applets 451, 613, 756
 APPLICATION 648, 666, 1566
 application/pdf 369, 1169, 1170
 Apply(); 528, 672
 appName 1167
 appVersion 1167
 Arbeitsbereich Breite 854, 1574
 Arbeitsbereich Breite des auf dem Bildschirm ohne Windows-Taskbar 1182, 1574
 Arbeitsbereich Höhe 854, 1574
 Arbeitsbereich Höhe auf dem Bildschirm ohne Windows-Taskbar 1182, 1574
 Arcus Cosinus 199, 1734
 Arcus Sinus 199, 1739
 Arcus Tangens 199, 1739
 AREA 455, 738, 1096
 area Objekt 455, 571, 739, 742
 area Objekt Events 1132, 1953
 area Objekt Filter 548
 area Objekt Styles 829
 areas Collection 455, 742
 Argument und Argumentenliste 103
 Argumente einer Funktion 103, 195, 1650
 Argumentenliste 40
 Argumentenliste Funktion 88, 106
 arguments JScript-Objekt 149
 arguments Objekt 149, 160
 arguments Script-Objekt 104, 160, 1567, 1589, 1650, 1739, 1745
 arithmetische Operatoren 75
 array Basis-Datenstruktur 54
 array Datentyp 54
 Array JScript Objekt VisualBasic-Array nach JScript-Array konvertieren 171
 Array JScript-Objekt 170, 1855, 1859, 1899, 1912
 Array JScript-Objekt 149
 Array JScript-Objekt Elemente anhängen 1859
 Array JScript-Objekt erstes Feldelement liefern und löschen 1899
 Array JScript-Objekt letztes Feldelement liefern und löschen 1855
 Array JScript-Objekt Werte dem Feld voransetzen 1912
 Array Konstruktor 164
 Array Objekt 149, 160
 Array Objekt aus Literalen 74
 Array Script-Objekt 54, 162, 166, 167, 168, 169, 172, 188, 1650, 1752, 1834, 1874, 1905, 1906
 Array Script-Objekt 160
 Array Script-Objekt Anzahl der Elemente 166, 1650
 Array Script-Objekt aus Literalen 174
 Array Script-Objekt Elemente anhängen 166, 1752
 Array Script-Objekt Feldelemente physisch sortieren 168, 1906

Array Script-Objekt komplett nach String konvertieren 167, 1834
 Array Script-Objekt Reihenfolge der Feldelemente physisch umkehren 167, 1874
 Array() 43, 44, 71
 Art der Objektpositionierung innerhalb Eltern 814, 817, 824, 1941
 Art des Buttons 599, 1721
 Art von Drag und Drop 1112, 1620
 Artist 874, 1469
 ASCII 57
 ASCII-Code der gedrückten Taste 1105, 1154
 ASCII-Code der Taste 1100, 1106, 1156
 ASCII-Code und Unicode 113
 ASCII-Zeichensatz 217, 988, 1775
 ASF 917, 928, 930, 972, 1107, 1131, 1497, 1726, 2000
 ASF-Datei 917, 928, 930, 972, 1107, 1131, 1497, 1726, 2000
 asiatische Zeichen Textfluss 811, 820, 1933
 asiatischen Zeichen Textabstand 815, 824, 1945
 ASP-Script 695
 Ast 222
 ASX 860, 861, 907, 913, 917, 920, 925, 926, 930, 934, 936, 940, 944, 949, 969, 970, 974, 1131, 1553, 1575, 1576, 1578, 1778, 1999
 ASX-Datei 860, 861, 907, 913, 920, 925, 926, 934, 936, 940, 944, 969, 970, 1131, 1553, 1575, 1576, 1578, 1778, 1999
 asynchrones Senden von Daten 1116, 1982
 ATOMICSELECTION 481, 560, 571, 589, 597, 609, 614, 618, 627, 648, 659, 666, 680, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 730, 743, 759, 774, 1014, 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1191, 1568
 Attribut automatisch erzeugen und mit Wert belegen 262, 484, 565, 570, 575, 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 Attribut einem Knoten zuweisen 238
 Attribut eines Knoten 263, 484, 565, 570, 575, 593, 601, 603, 612, 618, 622, 631, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 751, 752, 756, 763, 769, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1190, 1194, 1196, 1887
 Attribut entfernen 238
 Attribut entfernen anhand ID oder NAME 267, 467, 1868
 Attribut erzeugen 238
 Attribut erzeugen per HTML 238
 Attribut erzeugen per Script 238
 Attribut erzeugen und mit Wert belegen 262, 484, 565, 570, 575, 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 Attribut hinzufügen anhand Zeiger auf Attribut 268, 467, 1895
 Attribut HREF 455, 738
 Attribut im Dokument erzeugen 251, 431, 826, 1753
 Attribut systemnah 957



Attribut Wert 254, 262, 483, 484, 563, 564, 569, 570, 574, 575, 591, 593, 599, 601, 602, 603, 611, 612, 616, 618, 620, 621, 629, 650, 651, 661, 662, 664, 665, 668, 669, 683, 684, 689, 690, 693, 695, 698, 699, 701, 702, 705, 707, 710, 711, 715, 716, 719, 720, 723, 725, 728, 729, 732, 734, 737, 738, 740, 742, 746, 747, 750, 755, 756, 761, 763, 768, 769, 777, 778, 827, 828, 849, 850, 863, 864, 1021, 1027, 1030, 1031, 1034, 1035, 1038, 1039, 1044, 1045, 1049, 1050, 1054, 1056, 1060, 1062, 1066, 1068, 1072, 1074, 1080, 1082, 1172, 1189, 1190, 1193, 1194, 1777, 1886

Attribut Wirksamkeit 1624

Attribute des Objektes 246, 1693

Attribute eines Elementes 259, 437, 484, 564, 569, 574, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 698, 702, 706, 711, 715, 720, 724, 728, 733, 738, 741, 747, 755, 762, 768, 777, 827, 1023, 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081, 1189, 1194, 1838

Attribute eines HTML-Objektes 463

Attribute HTML 259, 484, 564, 569, 575, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 762, 768, 778, 827, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1866

Attribute mischen 238

attribute Objekt 255, 463, 464, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 751, 755, 761, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1196, 1778

Attribute Wirksamkeit im Dokument 426

ATTRIBUTENAME 878, 890, 896

attribute-Objekt 223, 238, 266

Attribute-Referenzen 464

attributes Collection 244, 463, 561, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1601

attributes Collection des DOM 266, 268, 275, 464, 468, 474

Attribut-Name 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661

Attributwert 222

Attribut-Wert 238

Attributwert Objekt 247, 599, 686, 688, 697, 701, 705, 709, 714, 718, 723, 727, 761, 767, 1078, 1727

Attribut-Zeiger liefern anhand ID oder NAME 267, 466, 1799

Audio 563, 737, 754, 874, 907, 1468, 1721

audio/basic 1556

audio/x-ms-wax 1470

audio/x-ms-wma 874, 1468

Audio-Stummschaltung 889, 1641

Audio-Visual Interleaved 671, 702

auf ActiveX prüfen 1375

auf Event-Eigenschaften prüfen 1143

auf Timeline pausieren 592, 1853

Aufbau Bookmark-Datei 862

Aufbau Favoriten-Datei 862

Aufgabe von HTML-Tags 224

Auflösung des Bildschirms in Breite 854, 1183, 1733

Auflösung des Bildschirms in Höhe 854, 1183, 1632

Auflösung Fenster ändern 299, 399, 1872

Aufruf einer Funktion 103

Aufruf einer Funktion als Wert eines HTML-Attributes 103

Aufruf einer Funktion mit Parameterliste 104

Aufruf Funktion 88, 106

Aufruf zyklisch 401, 1892

aufgerufen Eventhandler 294

Aufrufer einer Funktion 195, 1589

Aufwand beim Parsen 48, 61

Aufzählungsliste 985

Aufzählungsliste Elemente 811, 821, 1934

Aus- und Einblende von Bildern mit Bildwechsel 528, 672

Ausdruck 102

Ausdruck berechnen, aber den Wert nicht liefern 77

ausdrucken des Dokumentes 584

Ausführbarkeit von Javacode 1170, 1834

Ausführen eines Scriptcodes 211

ausführen Kommando 276, 434, 457, 459, 476, 771, 1084, 1768

ausführen Script 394, 1769

Ausführung Javascript-Funktion 40

Ausführung von Javascript 425, 1607

Ausgabe auf Printmedium 1655

Ausgabe-Datenstream schliessen 1749

Ausgabemedien-spezifische CSS-Datei importieren (@import) 982

Ausgabenmedien-spezifische Style-Sheet-Eigenschaften deklarieren 982

ausgewählter Eintrag in Auswahlliste 981

auslesen 639

Auslesen eines Eingabefeldes 639

auslösen Event 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 997, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1107, 1189, 1193, 1196, 1772

Ausnahmebedingung 192

Ausnahmebedingung private 192

Ausnahmetyp Run-Time-Error 194, 1659

Ausrichtung 609, 614, 666, 680, 703, 752, 759, 1014, 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069, 1562

Ausrichtung Objekt vertikal 816, 825, 1947

Ausrichtung Objekthalt 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1727

ausrücken 986

ausschneiden 1098

Ausschneide-Operation 1097

Ausschnitt 328, 985

Ausschnitt Koordinate links 849, 1593

Ausschnitt Koordinate oben 849, 1594

Ausschnitt Koordinate rechts 849, 1594

Ausschnitt Koordinate unten 849, 1593

Aussenrand Abstand links zum angrenzenden Objekt 812, 821, 1936

Aussenrand Abstand oben zum angrenzenden Objekt 812, 821, 1936

Aussenrand Abstand rechts zum angrenzenden Objekt 812, 821, 1936



Aussenrand Abstand unten zum angrenzenden Objekt 811, 821, 1936
 Aussenrand Abstand zur Umgebung 589, 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Aussenrand Objekt links, rechts, oben, unten 811, 821, 1935
 Aussenrand und Abstand zum Objekthalt 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Aus- und Einblende eines Bildes 528, 671
 Auswahlliste 981, 1097
 Auswertung von Pixelpositions-Angaben 984
 author 860, 861, 1778
 Author 748, 874, 1469
 auto 993
 Auto 1725
 Autocomplete 709, 727, 1731
 AutoComplete 1171
 AUTOCOMPLETE 635
 Autocomplete bei Formularen 1162
 AutoComplete Data Store 1163, 1742
 Autocomplete Löschen gespeicherter Werte 1573
 Auto-Layout Tabelle 815, 824, 1944
 automatisch scrollen 1876
 automatische Anfrage 1097
 automatischer Umbruch des Inhaltes des HTML-Elementes 992
 automatisches Kacheln 327
 automatisch-genriertes ID des Body 590, 615, 761, 767
 automatisch-genriertes ID des Objektes 239, 247, 430, 563, 568, 573, 599, 602, 619, 628, 649, 660, 663, 667, 682, 688, 692, 697, 701, 704, 709, 714, 718, 722, 727, 731, 737, 740, 745, 754, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1724
 autoReverse 1686, 1690
 Autoscan 1162
 Autoscan einer Url 1163, 1742
 AUTOSTART 581
 Autovervollständigung 622, 709, 727, 1163, 1171, 1731, 1738, 1742, 1763, 1778
 Autovervollständigung im Browser 1163, 1742
 Autovervollständigung im Formular 1162
 Autovervollständigung User-Profil lesen 1172, 1778
 Autovervollständigung vCard-Wert lesen 1172, 1778
 Autovervollständigung zum Formular 627, 708, 725, 1572
 AVI 671, 702
 back() 1165
 Back-Button 863, 1743
 BACK-Button 736, 1166
 BackColor 277, 476
 BackCompat 1595
 background-color 665
 Backslash 56, 58, 114
 Backslash Entwertung 56, 58, 114
 Backspace 56, 58, 114
 Balance 1574
 BALANCE 576
 Balance ganz links 581, 1574
 Balance ganz rechts 581, 1574
 Balance genau mittig 581, 1574
 Barn Filter 520
 barnDoorWipe 1703, 1723
 barWipe 1703, 1723

baseline 986, 1033, 1037, 1043, 1048, 1053, 1059, 1065, 1071, 1727
 BASEURL 862
 BasicImage Filter 514, 521, 2003
 Basis 10 199, 1651
 Basis 2 199, 1651
 Basis-Datenstruktur array 54
 Basis-Datenstruktur date 56, 176
 Basis-Datenstruktur function 56
 Basis-Datenstruktur Objekttyp 58
 Basis-Datenstruktur string 57
 Basis-Datenstrukturen 54
 Basis-Datentyp boolean 56, 175
 Basis-Datentyp Literal 56
 Basis-Datentyp number 57
 Basis-Datentyp Zeigertyp 58
 Basis-Datentypen 54
 Baumdarstellung 222
 Baumhierarchie der Objekte 222
 Baumhierarchie des Dokumentes 222
 bedingtes Parsen 53, 99
 BEGIN 867
 Behavior 490, 504
 Behavior .style.clientCaps 852
 Behavior .style.download 855
 Behavior .style.homePage 856
 Behavior .style.httpFolder 857
 Behavior .style.mediaBar 858, 859, 860, 861, 862, 1119, 1124, 1126, 1130, 1568, 1603, 1613, 1620, 1621, 1631, 1659, 1660, 1666, 1675, 1692, 1778, 1793, 1854, 1855, 1907, 1986, 1991, 1993, 1998
 Behavior .style.saveFavorite 862
 Behavior .style.saveHistory 863
 Behavior .style.time2 560, 561, 562, 563, 571, 572, 573, 575, 597, 598, 599, 609, 610, 614, 615, 618, 619, 627, 628, 666, 667, 681, 682, 686, 687, 688, 691, 692, 695, 696, 697, 700, 703, 704, 708, 709, 712, 713, 714, 717, 718, 721, 722, 725, 726, 727, 743, 744, 745, 765, 767, 774, 775, 776, 865, 890, 1076, 1077, 1078, 1107, 1114, 1117, 1120, 1121, 1124, 1125, 1127, 1128, 1129, 1131, 1191, 1192, 1468, 1555, 1557, 1559, 1560, 1562, 1568, 1573, 1579, 1583, 1584, 1586, 1588, 1590, 1593, 1603, 1606, 1616, 1617, 1619, 1621, 1623, 1626, 1628, 1629, 1630, 1631, 1632, 1641, 1642, 1646, 1647, 1648, 1649, 1655, 1656, 1657, 1658, 1666, 1669, 1670, 1673, 1675, 1676, 1680, 1681, 1686, 1690, 1691, 1693, 1698, 1703, 1705, 1706, 1707, 1709, 1713, 1718, 1719, 1723, 1725, 1726, 1728, 1729, 1730, 1732, 1734, 1736, 1743, 1760, 1764, 1765, 1845, 1847, 1850, 1851, 1853, 1855, 1857, 1871, 1873, 1877, 1878, 1879, 1880, 1882, 1904, 1979, 1984, 1987, 1988, 1991, 1992, 1994, 1995, 1996, 1997, 1999, 2000
 Behavior .style.time2 Import 876
 Behavior .style.time2 Medien zur Animation 874
 Behavior .style.time2 Time-Formate 872
 Behavior .style.time2 XML-Namensraum 875
 Behavior .style.userData 974
 Behavior browsereigene 505
 Behavior Customer-Tag 479
 Behavior des Internet Explorer 850
 Behavior dynamisch verwalten 1761
 Behavior eines Elementes 809, 819, 1919
 Behavior Element 507, 1761
 Behavior Objekt 507, 1761
 Behavior per *.htc-Datei und Event 1147
 Behavior Standard 479, 505, 809, 819, 1919



Behavior style.saveSnapshot 864
 Behavior style.time2 451, 593, 604, 773, 887
 Behavior style.userData 976, 1624, 1837, 1874
 Behavior time2 1713
 Behavior und Event 1147
 Behavior Url im Objekt Style 337, 781
 behavior: url(#default#time2 876
 Behavior-Collection .style.time2.playlist 934, 937, 939, 948, 951, 956, 1561, 1884
 Behavior-Datei 809, 819, 1919
 Behavior-Objekt .style.time2.animate 890
 Behavior-Objekt .style.time2.animateColor 896
 Behavior-Objekt .style.time2.animateMotion 900
 Behavior-Objekt .style.time2.animation 907
 Behavior-Objekt .style.time2.audio 911
 Behavior-Objekt .style.time2.excl 917, 940
 Behavior-Objekt .style.time2.img 920
 Behavior-Objekt .style.time2.media 924
 Behavior-Objekt .style.time2.playItem 934, 937
 Behavior-Objekt .style.time2.priorityClass 940
 Behavior-Objekt .style.time2.ref 944
 Behavior-Objekt .style.time2.set 949
 Behavior-Objekt .style.time2.switch 957
 Behavior-Objekt .style.time2.transitionFilter 508, 959, 965, 966, 1586, 1588, 1619, 1629, 1657, 1703, 1719, 1723, 1730
 Behavior-Objekt .style.time2.video 967
 Behavior-Objekt style.time2.par 930
 Behavior-Objekt style.time2.seq 952
 behaviorUrns Collection 504
 Beispiel Datenbank im Internet Explorer 1379
 Beispiel für 3D-Effekt eines Bildes 542
 Beispiel für Aus- und Einblende von Bildern mit Bildwechsel 528, 672
 Beispiel für Schatten eines Bildes 542
 Beispiel zur Tastatur-Eventbehandlung beim IE und Netscape 1156
 Beispiele für Aus- und Einblende eines Bildes 528, 671
 Beispiele für Style-Sheet 979
 Beispiele robots.txt 50
 Beispiele Style-Sheet 988
 Beispiele zur Eventbehandlung des Netscape 362
 Belegung Zeiger 70
 below 1016, 1627
 BELOW 326
 Benutzerdefinierter Tag zu einem HTML-Tag 663, 1734
 Benutzereingabe 635
 Benutzereinstellungen des Browsers 1168
 Benutzerinfo als Cookie 485
 Benutzer-Tag 663, 1734
 Berechnung der Punktnotation 48, 61
 Berechnung zur Zeitzone 176
 Bereich ab inklusive HTML-Start bis hinter -Ende-Tag 482, 561, 567, 572, 598, 610, 615, 619, 628, 659, 663, 665, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1029, 1033, 1037, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1666
 Bereich des Input-Objektes im Formular markieren 690, 694, 699, 706, 711, 716, 720, 725, 729, 1883
 Bereich des Objektes verlassen 1102
 Bereich zwischen HTML-Start und -Ende-Tag 481, 561, 597, 609, 618, 627, 659, 662, 665, 666, 730, 739, 744, 775, 1000, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1636, 1637

Bereich zwischen HTML-Start und -Ende-Tag Body 589, 614, 759, 766
 bereits geklickte Links Farbe 430, 1731
 Beschreibung einer Media-Datei 907, 913, 920, 925, 936, 944, 969, 1553
 Beschreibung Run-Time-Error 193, 1607, 1656
 Beschreibung zum Plugin 369, 1169, 1170
 Beschriftung eines Elementes 729
 Bestimmte Daten auf dem Server dürfen NICHT gescannt werden 50
 Bestimmte HTML-Dateien auf dem Server dürfen NICHT gescannt werden 50
 Bestimmte Verzeichnisse auf dem Server dürfen NICHT gescannt werden 50
 Betreff einer Email 626, 1559
 Betriebssystem 411, 1167, 1672
 Betriebssystem Name 1170, 1672
 Betriebssystem Sprache 1169, 1583
 Betriebssystem Sprache Installation 767, 776, 1170, 1708
 Betriebssystem Sprache laut Usereinstellung 854, 1727
 Betriebssystem Standardsprache 412, 854, 1708
 Betriebssystem Standard-Sprache laut Installation 767, 776, 1170, 1708
 Betriebssystem Standardsprache laut Usereinstellung 412, 1727
 Betriebssystem Windows 854, 1672
 Betriebssystem Informationen 1167
 betriebssystem-nahe Objekte 225
 Betriebssystem-Plattform und Browserversion 1169, 1567
 bewegen Textbereich 460, 1084, 1842
 Bezeichner der Variable 62
 Bezeichner des Events 1107, 1721
 Bezeichner des JScript-Objektes 61, 150, 209, 1595, 1596
 Bezeichner des Objektes 481, 561, 567, 572, 582, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 Bezeichner des Tag eines Objektes 247, 482, 562, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 665, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1708
 Bezeichner einer JScript-Objektklasse 61, 150, 209, 1595, 1596
 Bezeichner Objekttyp 61, 150, 209, 1595, 1596
 BGCOLOR 665, 809, 818, 1918
 BGSOUND 1229
 bgsound Objekt 575, 885, 890, 909, 915, 922, 928, 932, 946, 955, 972, 1658, 1732
 bgsound Objekt 873
 bgsound Objekt Events 1132, 1954
 bgsound Objekt mehrere im Dokument 576
 bgsound Objekt Styles 829
 bidirektionale Dateneingabe von Unicode 816, 818, 825, 1947
 Bild 907, 1101
 Bild animieren 508
 Bild animiert 1101
 Bild Hintergrund Position 809, 819, 1919
 Bild HTML-Container 671



Bild im Hintergrund 809, 818, 1918
 Bild Lade-Ereignisse 1161
 Bild laden 129, 1100, 1147
 Bild vollständig geladen 1161
 Bild vorladen 675
 Bild zum Microsoft Active Desktop hinzufügen 1162, 1737
 Bildanimation 508
 Bildausschnitt 993
 Bildausschnitt maximieren 993
 Bilder in den RAM laden 674
 Bilder mit Bildwechsel 528, 672
 Bild-Ladezustand 671
 Bildpunkt 1182
 Bildpunkte horizontal 1183, 1607, 1652
 Bildpunkte vertikal 1183, 1608, 1652
 Bildschirm 1182, 1574, 1655
 Bildschirm Anzahl der horizontalen Bildpunkte 1183, 1652
 Bildschirm Anzahl der vertikalen Bildpunkte 1183, 1608
 Bildschirm Auflösung in Breite 1183, 1733
 Bildschirm Auflösung in der Höhe 854, 1632
 Bildschirm Auflösung in Höhe 1183, 1632
 Bildschirm Fontglättung 1183, 1627
 Bildschirm off-screen bitmap buffer 1182, 1583
 Bildschirm Refresh-Intervall 1183, 1725
 Bildschirmauflösung 1182
 Bildschirms Auflösung in Breite 854, 1733
 Bildschirm Anzahl der horizontalen Bildpunkte 1183, 1607
 Bildschirm Anzahl der vertikalen Bildpunkte 1183, 1652
 Bildverarbeitung 508
 Bildverarbeitungsprogramm 508
 Bildwechsel 528, 672
 binärer Code 1761
 Bitmaske für Zustand der Steuertasten 1105, 1156
 Bitoperatoren 76
 Bitrate 874, 1469
 Bits pro Pixel für eine Farbe 1182, 1583, 1594
 Bitverschiebung 79, 80
 Bitverschiebung Operator 78
 Bitverschiebungen 76
 bitweise Operatoren 76
 Blank 144
 Blank geschützt 816, 826, 1948
 Blattvorschub 56, 58, 114
 Blau-Anteil 980
 BlendTrans Filter 522
 blind copy 626, 1559
 Blinds Filter 523
 blink 815, 817, 824, 987, 990, 1945
 Blockanweisung 85
 Blockelement 809, 818, 1917
 Blockformat 277, 476
 Blocksatz 815, 825, 986, 990, 1946
 Blur Filter 514, 523, 2004
 blur() 638, 1634
 BMP 671, 702
 BODY 223, 752, 1095, 1097, 1231
 Body Abstand von linken sichtbaren Randes des Umgebungsobjektes 590, 615, 760, 766
 Body Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen 590, 615, 760, 766
 Body aktueller Status 590, 615, 760, 766
 Body Bottom Margin 589, 1582

Body Breite des horizontalen Scrollbereiches 590, 615, 760, 766
 Body Content-Editierbarkeit 589, 614, 759, 766
 Body durch den Browser automatisch-genriertes ID 590, 615, 761, 767
 Body Eltern 482, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1195, 1667
 Body Farbe eine VLINK 590
 Body Farbe eines ALinks 589
 Body Farbe eines Links 590
 Body Focussierbarkeit 589, 614, 759
 Body Hintergrundbild 589
 Body Hintergrundbild Scrollbarkeit 589
 Body Höhe des vertikalen Scrollbereiches 590, 615, 760, 766
 Body ID 589, 614, 759, 765
 Body Index des Elementes in der Tab-Tasten-Folge 590, 615, 760
 Body Index in der Collection document.all 590, 615, 760
 Body Inline-Style 590, 615, 760, 766
 Body Klassenreferenz 589, 614, 759, 765
 Body Knotentyp 590, 614, 760, 766
 Body Knotenwert 590, 614, 760, 766
 Body Left Margin 590, 1650
 Body Mehrzeiligkeit Inhalt 589, 614, 759, 766
 body Objekt 456, 457, 458, 504, 770, 771, 772, 864, 866, 885, 887, 890, 894, 895, 898, 900, 905, 906, 908, 909, 911, 914, 915, 917, 921, 922, 924, 927, 928, 930, 932, 934, 945, 946, 948, 949, 950, 951, 954, 955, 957, 966, 967, 971, 972, 974, 1082, 1125, 1128, 1658, 1709, 1732, 1883, 1992, 1995
 body Objekt Events 1132, 1954
 body Objekt Filter 548
 body Objekt Styles 829
 Body Referenz auf das Elternobjekt 482, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1195, 1667
 Body Referenz auf das Vorgängerkind 590, 615, 760, 766
 Body Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag 589, 614, 759, 766
 Body Referenz der Eltern 590, 615, 760, 766
 Body Right Margin 590
 Body Scrollbar-Anzeige 590
 Body Selektierbarkeit 589, 590, 614, 615, 759, 761
 Body Sprache für Anzeige von Sonderzeichen etc. 589, 614, 759, 766
 Body Sprache für Script 590, 614, 759, 766
 Body Status 590, 615, 760, 766
 Body Tag-Bezeichner 590, 615, 761, 767
 Body Tastaturzugriff 589, 614, 759
 Body Textbereich des Elternobjektes 590, 615, 760, 766
 Body Textfarbe 590, 1711
 Body Tooltip-Text 590, 615
 Body Top Margin 590
 Body Umfluss 589, 614
 Body Umflussrichtung 589, 614, 759, 765
 Body Vordergrundfarbe 590, 1711
 Body Wortumbruch 590



Body X-Koordinate der linken oberen Ecke 590, 615, 760, 766
 Body X-Koordinate der rechten unteren Ecke 590, 615, 760, 766
 Body Y-Koordinate der linken oberen Ecke 590, 615, 760, 766
 Body Y-Koordinate der rechten unteren Ecke 590, 615, 760, 766
 Body Zeiger auf das NACHFOLGENDE Kind 590, 614, 760, 766
 body.focus() 1775
 BODY-Abschnitt 40
 Body-Content-Editierbarkeit 589, 614
 body-Events 1209, 1965
 body-Objekt 665
 BODY-Objekt 223, 424, 1560
 Bogenmass 199, 1734
 bold 987, 988, 989
 Bold 277, 476
 bolder 987, 989
 Bookmark 277, 278, 476, 477, 862, 1129, 1996
 Bookmark im Textbereich 459, 1084, 1778
 Bookmark Textbereich 460, 1084, 1842
 Bookmark-Datei Aufbau 862
 Bookmarks 1108
 bookmarks Collection 1108
 boolean 54, 57, 77, 81, 143
 Boolean 1376
 boolean Basis-Datentyp 56, 175
 boolean Datentyp 56, 175
 Boolean erzeugen 152, 1745
 Boolean JScript-Objekt 149
 Boolean Objekt 149, 160
 Boolean Script-Objekt 56, 160, 175
 boolean_instanz 175
 border 671, 1016, 1627
 Border 983
 BORDER 325, 670, 671
 Borderdicke 659, 666, 681, 1014, 1581
 Borderfarbe 648, 659, 1014, 1057, 1063, 1069, 1581
 both 811, 821, 1933
 bottom 986, 1029, 1033, 1037, 1043, 1048, 1053, 1059, 1066, 1071, 1727
 Bottom Margin Body 589, 1582
 BOUNDARY 907, 913, 920, 926, 944, 970, 1582
 box 1016, 1627
 Box 277, 476, 614
 break Anweisung 85, 92, 96
 break-word 990
 Breite des Arbeitsbereiches 854, 1574
 Breite des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar 1182, 1574
 Breite des horizontalen Scrollbereiches 482, 562, 568, 598, 610, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 Breite des horizontalen Scrollbereiches Body 590, 615, 760, 766
 Breite des Objektes 682, 745, 754
 Breite Frame 659, 1595
 Breite in Pixel vom Fenster 1106
 Breite in Pixel vom Frame 1106
 Breite Objekt 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 813, 814, 816, 817,

823, 824, 826, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592, 1940, 1941, 1948
 Browser 130, 1100, 1147
 Browser Autocomplete einer Url 1163, 1742
 Browser Autoscan einer Url 1163, 1742
 Browser Autovervollständigung 1163, 1742
 Browser Autovervollständigung einer Url 1163, 1742
 Browser bedingtes Parsen 53, 99
 Browser Betriebssystem 411, 1169, 1567
 Browser Betriebssystem Standardsprache 412
 Browser Codename 410, 1170, 1566, 1726
 Browser eigene Behavior 505
 Browser erzeugt Zeiger 59
 Browser Haupt-Versionsnummer 411, 1567
 Browser HTTP User-Agent 412
 Browser Informationen 410, 1167
 Browser interner Zeiger 59
 Browser interpretiert 222
 Browser Javacode 1170, 1834
 Browser Java-Maschine 1170, 1834
 Browser Java-Verfügbarkeit 412
 Browser kennt FRAMESET nicht 750
 Browser kennt Script nicht 40
 Browser kennt SCRIPT-Tag nicht 751
 Browser Majorversion 411, 1567
 Browser Name 1169, 1566
 Browser Nutzbarkeit von Cookie 411, 854, 1598
 Browser Objektfähigkeit 136
 Browser Onlinestatus 1170, 1665
 Browser Parsen bedingt 53, 99
 Browser parst 222
 Browser Performance 876
 Browser Plattform 1169, 1567
 Browser plugin-fähig 1171
 Browser Speichermangel 114
 Browser Standard-Behavior 505
 Browser Standardsprache des Betriebssystems 412
 Browser Stop-Button 1104
 Browser und Aufwand beim Parsen 48, 61
 Browser und Cookies 486
 Browser und Objekte 131
 Browser und Scriptmaschine 48, 61
 Browser User-Agent 412
 Browser Verfügbarkeit von Java 412
 Browser Version 1170, 1567, 1726
 Browser Versionsnummer 410, 1566
 Browser vordefinierte Objekte 220
 Browser Zeiger intern 59
 Browser-Betriebssystem 1167
 Browser-Cache 307, 1165
 Browser-Codename UND -Version 1168
 Browsereinstellung 294
 Browser-Einstellungen zu Sicherheit 1162
 Browsererkennung 36
 Browsererkennung anhand der Unterscheidung browserinterner Objekte 37
 Browsererkennung beim Internet Explorer 38
 Browsererweiterungen 1375
 Browser-Erweiterungen Internet Explorer 749
 Browserfähigkeiten 223, 1145
 Browserfenster 281, 283, 327, 370, 1104
 Browserfenster rausschieben und danach neu anzeigen 294, 330, 395, 1839
 Browserfenster schliessen 1104
 Browserfenster Statuszeile 1637



Browserfenster Status-Zeile 1637
 Browserfenster und HTML-Dokument 220
 Browserfensterecke 1182
 Browserhersteller 232
 browserhersteller-spezifische Erweiterungen 223
 Browser-Hilfe 1100
 Browserinstanz 1108
 browserinterne Objekte 224
 browserinternes Objekt 37, 68
 Browsername 410, 1167, 1567
 Browserperformance 48
 Browser-Performance 48, 61
 Browserperformance und Javascript 48
 Browserperformance und JScript 48
 Browserperformance und Punktnotation 48, 61
 Browserperformance und Script 48
 Browserperformance und Zeiger 61
 Browser-Plattform 410
 Browserprüfung 136
 Browsers Benutzereinstellungen 1168
 Browsers Codename 1169, 1566
 Browser-Search-Fenster 396, 438
 Browsersprache 411, 1583
 Browser-Sprachversion 1167
 Browserversion 232, 410, 1167
 Browser-Version 17
 Browserversion Betriebssystem 1169, 1567
 Browserversion Minorversion 1169, 1566
 Browserversion Plattform 1169, 1567
 Browserversion Update 1169, 1566
 Browserversionen 136
 Browser Sicherheitseinstellungen 1162
 Buchstabe Style 809, 818, 1917
 Buildnummer der Jscript-Scriptmaschine 100
 Bullet 985
 button 632, 685, 1721, 1722
 -Button 622
 BUTTON 1095, 1096, 1647
 Button Art 599, 1721
 Button Back 863, 1743
 Button Formular 593
 Button Forward 863, 1775
 Button im Dialogfenster 981
 button Objekt 457, 458, 593, 622, 632, 685, 686, 770, 771, 1082, 1883
 button Objekt Events 1133, 1954
 button Objekt Filter 549
 button Objekt Styles 830
 Button Submit für Formular 721
 Button Vorwärts 863, 1775
 Button Zurück 863, 1743
 Button-Control 277, 476, 686, 1722
 Button-Control Reset 1722
 Button-Control Sumbit 1722
 Button-Text im Dialogfenster 981
 bzeichenfolge 1184
 Bzeichenfolge 1184
 Cache 1165
 cancelBubble 1102
 canHaveHTML 242, 481, 560, 567, 571, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 730, 736, 739, 743, 750, 751, 752, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1069, 1076, 1187, 1191, 1195, 1590
 canSlip 1705

capitalize 987, 990
 CAPTION 1019, 1027, 1339, 1754, 1758
 captureEvents() 361, 1146
 carbon copy 626, 1559
 Carriage return 56, 58, 114
 Cascading Style Sheets Sprache 826, 1723
 catch 193
 CD Table of Contents Identifier 874, 1469
 CD TOC 874, 1469
 CDATA-Section 239
 Ceckbox-Control 1078, 1699
 center 406, 408, 986, 990, 1562
 centerTop 1704, 1723
 CGI-Script 695
 Channel Definition Format-Datei 1164, 1830
 Channel hinzufügen 1162, 1737
 Channelleiste 397, 438
 channelmode 396, 438
 Channel-Mode anzeigen 397, 438
 Charakter-Set 425, 1606
 Charset des Dokumentes 626, 1556
 checkbox 633, 685, 1722
 Checkbox 1097
 CHECKBOX 1095, 1096
 Checkbox abgehakt 634
 Checkbox-Control 690, 1722
 Checkbox-Control Grauzustand (Dimmed) und Selektiertheit 691, 1635
 Checkbox-Control Selektionsstatus 691, 712, 713, 1591, 1606
 checked 637
 CHECKED 633, 634, 635, 636, 712
 CheckerBoard Filter 524
 Child 222
 childNodes Collection 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 childNodes-Collection 241
 children Collection 452, 454, 632, 641, 685, 707
 Chroma Filter 525
 circ 1688
 circle 571, 1598, 1688, 1704, 1723
 CLASS 339, 605, 774, 783, 807, 876, 979, 999
 CLASS nicht gefunden 66, 114
 CLASS-Attribut und ID-Attribut-Wert 805
 CLASSID 752
 clear() 431, 1746
 Click auf das Radiobutton 637
 Click auf Scrollbar simulieren 591, 611, 777
 click() 638
 Click-Event 1149
 Client 738
 clientCaps 411, 412, 1665, 1834
 clientInformation 1235
 CLIP 327, 328
 Clip Position sichtbarer Bereich über dem Objekt 810, 820, 1928
 Clipboard 277, 278, 413, 476, 477, 1097, 1103, 1108, 1112, 1114, 1115, 1160, 1978, 1979, 1981
 Clipboard auslesen 416, 1112, 1781
 Clipboard füllen 416, 1112, 1887
 Clipboard hinzufügen 1115, 1981
 clipboardData Objekt 378, 1108, 1593
 Clipboardinhalt löschen 416, 1112, 1746



Clipfenster 810, 820, 1928
 Clippingregion 849, 1593, 1594
 clockWipe 1704, 1723
 clockwiseTwelve 1704, 1723
 CLSID:05589FA1-C356-11CE-BF01-00AA0055595A
 1287
 clsid:333C7BC4-460F-11D0-BC04-0080C7055A83
 1001, 1376
 CODE 306, 450, 566, 570
 CODEBASE 306, 450, 566, 570
 Codename Browser 1170, 1726
 Codename des Browsers 410, 1169, 1566
 col 1065, 1071, 1683
 COL 1032
 colgroup 1065, 1071, 1683
 COLGROUP 1035
 Collection.all 1196
 Collection.anchors 1197
 Collection.applets 1197
 Collection.areas 1198
 Collection.attributes 1198
 Collection.behaviorUrns 1198
 Collection.blockFormats 1199
 Collection.boundElements 1199
 Collection.cells 1199
 Collection.childNodes 1199
 Collection.children 1200
 Collection.controlRange 1200
 Collection.document.all 1201
 Collection.elements 1201
 Collection.embeds 1201
 Collection.filters 1201
 Collection.fonts 1202
 Collection.forms 1202
 Collection.frames 1202
 Collection.images 1203
 Collection.imports 1203
 Collection.links 1203
 Collection.namespaces 1203
 Collection.options 1204
 Collection.pages 1204
 Collection.plugins 1204
 Collection.rows 1204
 Collection.rules 1205
 Collection.scripts 1205
 Collection.styleSheets 1205
 Collection.tBodies 1206
 Collection.TextRange 1206
 Collection.TextRectangle 1206
 Collection.timeChildren 451, 593
 Collection aller im Dokument befindlichen Objekte 256,
 435, 629, 827, 1786
 Collection Anzahl Elemente 451, 453, 454, 455, 511,
 593, 642, 652, 685, 707, 738, 742, 940, 998, 999,
 1000, 1040, 1046, 1165, 1650
 Collection areas 455, 742
 Collection attributes 244, 463, 561, 567, 572, 590, 598,
 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691,
 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739,
 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036,
 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191,
 1601
 Collection attributes des DOM 266, 268, 275, 464, 468,
 474
 Collection behaviorUrns 504
 Collection bookmarks 1108

Collection childNodes 243, 561, 567, 572, 597, 601,
 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696,
 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759,
 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052,
 1058, 1064, 1070, 1077, 1188, 1191, 1627
 Collection children 452, 454, 632, 641, 685, 707
 Collection document.all 447, 453, 456, 482, 598, 602,
 628, 648, 649, 651, 660, 663, 665, 666, 667, 682, 687,
 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 736,
 740, 745, 750, 754, 757, 763, 769, 1000, 1018, 1029,
 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1188,
 1597, 1692
 Collection document.all Index des Objektes 610, 775
 Collection document.anchors 448, 565
 Collection document.applets 450, 570
 Collection document.body.timeAll 451, 593, 865
 Collection document.cookie 488, 974
 Collection document.embeds 451, 613, 756, 1170, 1171
 Collection document.form.elements 452, 641
 Collection document.forms 452, 641
 Collection document.frames 382, 453, 648, 651, 658,
 665, 1627, 1651
 Collection document.images 454, 671, 684, 707
 Collection document.links 455, 738
 Collection document.namespaces 504, 506, 507
 Collection document.scripts 1190
 Collection document.select.options 456, 757, 760, 763,
 769, 1666, 1687
 Collection document.selection.controlrange 456, 770
 Collection document.selection.controlRange 458, 770,
 1083, 1757, 1764
 Collection document.selection.textrange 457, 458, 770,
 771, 1083, 1757, 1764
 Collection document.styleSheets 994, 999
 Collection Element entfernen 456, 457, 742, 750, 762,
 770, 771, 1865
 Collection event.bookmarks 1108
 Collection FileSystemObject.Drives 1435
 Collection FileSystemObject.Folder.Files 1440, 1441,
 1625
 Collection FileSystemObject.Folder.Folders 1441, 1442,
 1703
 Collection filters 511
 Collection frames 665
 Collection history 1165
 Collection ID_Player.cdromCollection 1483
 Collection ID_Player.mediaCollection 1491
 Collection ID_Player.playlistCollection 1494
 Collection namespaces 1761
 Collection navigator.mimeType 451, 613, 756, 1168,
 1169, 1170, 1171
 Collection navigator.plugins 311, 1168, 1169, 1170,
 1171
 Collection playList 911, 913, 914, 925, 926, 927, 940,
 944, 945, 970, 971, 1575, 1576, 1578, 1632, 1635
 Collection plugins 451, 613, 756
 Collection styleSheet.imports 997, 1738
 Collection styleSheet.pages 998, 1738
 Collection styleSheet.rules 998, 1738, 1868
 Collection table.rows 1039, 1059, 1682
 Collection table.rows.cells 1040, 1063, 1069, 1590
 Collection table.tBodies 1046
 Collection table.tBody.rows 1045, 1059, 1065, 1685
 Collection table.tBody.rows.cells 1045
 Collection table.tFoot.rows 1051, 1059, 1065, 1685
 Collection table.tFoot.rows.cells 1051



Collection table.tHead.rows 1056, 1059, 1065, 1685
 Collection table.tHead.rows.cells 1056
 Collection TextRange.TextRectangle 458, 460, 1082, 1085
 Collection timeChildren 865, 890
 Collection verwalten 188
 Collection window.document.anchors des Netscape 306
 Collection window.document.applets des Netscape 306
 Collection window.document.cookie des Netscape 307
 Collection window.document.embeds des Netscape 309
 Collection window.document.forms des Netscape 309
 Collection window.document.frames des Netscape 310
 Collection window.document.images des Netscape 310
 Collection window.document.layers des Netscape 310
 Collection window.document.links des Netscape 311
 Collection window.document.plugins des Netscape 311
 Collection window.navigator.mimeTypes des Netscape 368, 369
 Collection window.navigator.plugins des Netscape 368, 369
 Collectionen 223
 Color Hintergrund 809, 818, 1918
 cols 1017, 1683
 Comma Separated Values 1376
 command Objekt 275, 474
 comment Objekt 53, 601, 1650
 Comment Objekt 252, 432, 1754
 comment Objekt Events 1133, 1954
 comment Objekt Styles 830
 Comment-Objekt 238
 complete 671
 Compositor Filter 525
 const 101
 Container 222, 251, 483, 563, 569, 574, 591, 599, 611, 616, 620, 629, 650, 660, 664, 668, 683, 688, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 750, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1753
 contains() 241
 CONTENT 748, 863, 864
 Content vom Objekt Editierbarkeit 481, 560, 589, 597, 609, 614, 618, 627, 686, 708, 713, 717, 721, 725, 730, 743, 774, 1076, 1191, 1596
 Content-Editierbarkeit Body 589, 614, 759, 766
 continue Anweisung 86, 92
 control 1723
 Control 277, 434, 457, 459, 476, 771, 1084, 1768
 Control für mehrzeilige Text-Eingabe 1074
 Control für Text-Eingabe 1074
 Control für Text-Eingabe mehrzeilig 1074
 CONTROL_MASK 358, 363, 364, 1105, 1156, 1157
 Control-Element 688, 692, 697, 699, 700, 704, 709, 714, 718, 722, 727, 770, 1607, 1721
 Control-Element Selektionsstatus 692, 714, 1078, 1591, 1699
 Control-Elemente 456, 457, 770, 771, 1883
 Control-Elemente einer Selektion 456, 770
 Control-Elemente Selektion 457, 771, 1883
 Controlrange Selektion 457, 771, 1883
 ConversionError Run-Time-Error 194, 1659
 Cookie als Benutzerinfo 485
 Cookie als Formulardate 485
 Cookie als Password 485
 Cookie als Umgebungseinstellung 485
 Cookie Anwendungsbereich 488
 Cookie des Dokumentes 424, 1597

Cookie Eigenschaften 488
 Cookie Feld 177, 425, 489, 1598, 1781, 1784, 1789, 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809, 1811, 1814, 1816, 1818, 1820, 1822
 Cookie lesen 487
 Cookie löschen 489
 Cookie Name 177, 425, 489, 1598, 1782, 1784, 1789, 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809, 1812, 1814, 1816, 1818, 1820, 1822
 Cookie Nutzbarkeit im Browser 411, 1598
 Cookie path 488
 Cookie schreiben 486
 Cookie Seperator 177, 425, 489, 1598, 1782, 1784, 1789, 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809, 1811, 1814, 1816, 1818, 1820, 1822
 Cookie Servergruppezugehörigkeit 488
 Cookie Verfallsdatum 489
 Cookie verwalten (Beispiele) 486
 Cookie-Ersatz durch input hidden Objekt 699
 Cookienutzbarkeit im Browser 854, 1598
 Cookies 699
 Cookies Abschaltung 486
 Cookies Lage auf dem PC 486
 Cookies löschen 486
 Cookies und Browser 486
 Cookies und Firewall 486
 Cookies und Surfgeohnheiten des Users 485
 Cookies Vermeidung 486
 Cookies Verwaltung 486
 Cookies Verwendung 484
 Cookies Zweck 484
 Cookiesstatus 1170, 1598
 Cookiesverwaltung 1167
 Cookie-Verwaltung 484
 Cookie-Verwaltung im Netscape 307
 Cookiewert 488
 COORDS 571, 1598
 copy 1617
 Copy 277, 476, 1115, 1981
 copyLink 1620
 copyMove 1620
 copyright 860, 861, 1778
 Copyright 874, 1469
 Copyright der Media-Datei 908, 914, 921, 926, 936, 944, 970, 1599
 CopyRight-Meldung 313, 320, 643, 654
 Cosinus 199, 1734, 1753
 CPU Informationen 1167
 CPU-Hersteller 411, 854, 1601
 CPU-Klasse 1170, 1601
 createAttribute() 259, 564, 569, 575, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 827, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1866
 CreateBookmark 277, 476
 CreateLink 277, 476
 createTextRange() 458, 591, 599, 688, 701, 710, 719, 723, 728, 1079, 1082, 1757
 CreationDate 874, 1469
 crossfade 1704, 1723
 crosshair 988, 993
 CSS 223, 797, 847, 849, 976, 1100
 CSS des Netscape 334
 CSS Kompatibilität des IE 6.x zu CSS1 424, 1595
 CSS Sprache 826, 1723



CSS Überschrift mit permanentem Farbwechsel 1928
 CSS und Attribute width und height 803
 CSS1 801
 CSS1- Inkonformität des Internet Explorer 801
 CSS1- Kompatibilität auch bei Frameset-Dokument 802
 CSS1- Kompatibilität außer in einem Frameset-Dokument 802
 CSS1 Kompatibilität des IE 6.x zu CSS1 424, 1595
 CSS1- Konformität genau im Umfang von HTML 4 laut World Wide Web Consortium 802
 CSS1Compat 1595
 CSS1-Konformität Internet Explorer 801
 CSS2 801
 CSS-Attributwert 1601
 CSS-Datei 994
 CSS-Datei importieren 996, 1738
 CSS-Datei Url 996, 1634
 CSS-Klassen 796, 976
 CSS-Konformität der Versionen Internet Explorer 801
 CSS-Standard 801
 CSV-Format 1376
 Ctrl 1105
 CTRL-C 1108, 1115, 1981
 CTRL-F 1164, 1844
 CTRL-Taste 1153
 Ctrl-Taste links 1104
 CTRL-Taste links Status 1107, 1602
 CTRL-Tasten-Status 1107, 1601
 CTRL-V 1108
 CTRL-X 1108
 cueing 866
 current 1165
 currentStyle Objekt 797, 847, 849, 1593, 1594
 currentStyle Objekt Styles 830
 currentStyle Objekt Filter 549
 currTimeState Objekt 560, 561, 562, 563, 571, 572, 573, 597, 598, 599, 604, 609, 610, 614, 615, 618, 619, 627, 628, 666, 667, 681, 682, 686, 687, 688, 691, 692, 695, 696, 697, 700, 703, 704, 708, 709, 712, 713, 714, 717, 718, 721, 722, 725, 726, 727, 743, 744, 745, 765, 767, 773, 774, 775, 776, 887, 890, 943, 1014, 1018, 1028, 1029, 1041, 1042, 1046, 1048, 1051, 1053, 1057, 1059, 1063, 1065, 1069, 1071, 1076, 1077, 1078, 1107, 1114, 1117, 1120, 1121, 1124, 1125, 1127, 1128, 1129, 1131, 1191, 1192, 1553, 1555, 1557, 1559, 1560, 1562, 1568, 1573, 1579, 1583, 1584, 1586, 1588, 1590, 1593, 1603, 1606, 1616, 1617, 1619, 1621, 1623, 1626, 1628, 1629, 1630, 1631, 1632, 1633, 1636, 1641, 1642, 1645, 1646, 1647, 1648, 1649, 1654, 1655, 1656, 1657, 1658, 1666, 1669, 1670, 1671, 1673, 1675, 1676, 1680, 1681, 1686, 1690, 1691, 1693, 1696, 1698, 1703, 1705, 1706, 1707, 1709, 1711, 1713, 1718, 1719, 1723, 1725, 1726, 1728, 1729, 1730, 1732, 1734, 1736, 1742, 1743, 1760, 1764, 1765, 1845, 1847, 1850, 1851, 1853, 1855, 1857, 1871, 1873, 1877, 1878, 1879, 1880, 1882, 1904, 1979, 1984, 1987, 1988, 1991, 1992, 1994, 1995, 1996, 1997, 1999, 2000
 Cursor auf das Button setzen 633
 Cursor Maus Art 810, 820, 1930
 Cursor Standard 1617
 Cursor vom Button entfernen 633
 Cursorform 337, 781, 828, 1912, 1930, 1952
 Cursorform als Datei 337, 781, 828, 1912, 1930, 1952
 Cursorform im Browser implementiert 337, 781, 828, 1912, 1930, 1952

Cursorgeschwindigkeit 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 660, 664, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751
 Cursor-Layout bei Drop 1112, 1617
 custom Objekt 479, 801
 custom Objekt Events 1133, 1954
 custom Objekt Filter 549
 custom Objekt Styles 830
 Customer-Tag 479
 Cut 277, 476
 Cut & Paste 1104
 dashed 983, 987, 991
 DATA 369, 1169, 1170
 Data Stream 907, 908, 913, 914, 920, 921, 926, 927, 944, 945, 970, 971, 1583, 1646
 Data Tainting-Verfügbarkeit 412, 1909
 DATAFLD 1108
 DATAFORMATAS 1108
 Datasource-Objekt 569, 753, 755, 1195, 1196, 1679, 1842
 DATASRC 1108, 1603
 Data-Tainting 1169, 1170, 1909
 Date 748, 1376
 date Basis-Datenstruktur 56, 176
 date Datentyp 56, 176
 Date JScript-Objekt 149
 Date Objekt 149, 160
 Date Script-Objekt 56, 160, 176, 1781, 1783, 1788, 1791, 1793, 1795, 1797, 1800, 1804, 1807, 1809, 1811, 1813, 1815, 1818, 1820, 1822, 1851, 1888, 1892, 1894, 1895, 1898, 1899, 1909, 1910, 1911, 1912, 1913
 Date Script-Objekt Datum-Operation 180, 181, 182, 183, 184, 185, 186, 187, 1781, 1783, 1788, 1791, 1793, 1795, 1797, 1800, 1802, 1804, 1807, 1809, 1811, 1813, 1815, 1818, 1820, 1822, 1851, 1888, 1892, 1894, 1895, 1898, 1899, 1909, 1910, 1911, 1912, 1913
 Date Script-Objekt Zeitzone 181, 1804
 Datei 45
 Datei *.icm 511, 2001
 Datei als Textstream 1442
 Datei anlegen und öffnen 1431, 1757
 Datei Anzahl der Zeilen 1444, 1651
 Datei Attribute 1437, 1570
 Datei Datum des letzten Updates 681, 1626
 Datei Download 559, 855, 856, 1906
 Datei drucken 398, 1859
 Datei Elternordner 1438, 1668
 Datei HTML Component (HTC) 850
 Datei komplett lesen 1443, 1568
 Datei kopieren 1431, 1438, 1753
 Datei laden 855
 Datei lesen in Schleife 1443, 1568
 Datei lesen komplett 1444, 1860
 Datei lesen zeichenweise 1444, 1860
 Datei lesen zeilenweise 1444, 1860
 Datei löschen 1432, 1438, 1758, 1759
 Datei mit Style-Sheet-Informationen einbinden (*.css) 981
 Datei öffnen 1433, 1438, 1850
 Datei schliessen 1444, 1751
 Datei schreiben zeichenweise 1445, 1913
 Datei schreiben zeilenweise 1445, 1914
 Datei und Pfad eines Objektes 562, 572, 1167, 1671



Datei verschieben 1433, 1438, 1839
 Datei Zeilenumbruch 1445, 1914
 Datei Zeilenwechsel 1445, 1914
 Dateiänderung Datum 1437, 1606
 Dateiartern Image 671
 Dateiartern Video 671
 Dateien kopieren 1431, 1753
 Dateien löschen 1432, 1759
 Dateien verschieben 1433, 1839
 Dateierstellung Datum 1437, 1605
 Dateigröße in Bytes 1438, 1692
 Datei-Menü 398, 1859
 Dateiname 45, 634, 1438, 1660
 Dateiname 8.3-Kodierung 48
 Dateiname als 8.3 Bezeichner 1438, 1689
 Dateiname mit Suffix 1433, 1788
 Dateiname zufällig erzeugen 1433, 1802
 dateiname#hashtext 1166
 dateiname?searchtext 1166
 Dateiname-Teil einer Url ohne Path und ohne Protokoll 681, 1660
 Dateipfad 24, 1432, 1438, 1671, 1778
 Dateipfad absolut 24
 Dateipfad als 8.3 Pfad 1438, 1690
 Dateipfad relativ 24
 Dateisuffix 1433, 1438, 1724, 1788
 Dateisuffixe zum Plugin 369, 1169, 1171
 Dateisystem in JScript 1430
 Dateisystem Zugriff per JScript 1430
 Dateiverzeichnis lesen 1559, 1656
 Dateizugriff Datum 1437, 1606
 Daten an Plugin übergeben 369, 1169, 1170
 Daten asynchron senden 1116, 1982
 Daten des Objektes 601, 1603
 Daten Drag und Drop 1108
 Daten eines Formulars permanent speichern 1163, 1742
 Daten eines Formulars speichern 1163, 1742
 Daten eines Plugins 451, 613, 756, 1171
 Daten frei definierbar nicht permanent speichern 863, 864
 Daten frei definierbar permanent speichern 862
 Daten gekapselt 223
 Daten nicht permanent speichern 863, 864
 Daten permanent speichern 862, 1162, 1163, 1742
 Daten speichern 1163, 1742
 Daten über mehrere Onlinesitzungen speichern 862
 Daten Url 568, 649, 667, 682, 704, 1188, 1195, 1694
 Datenaustausch zwischen Frames 316, 322, 646, 656
 Datenbank im Internet Explorer 1376
 Datenbeschaffung formular-orientiert 622
 Dateneingabe von Unicode bidirektional 816, 818, 825, 1947
 Datenfeld 598, 610, 649, 667, 682, 692, 700, 709, 713, 726, 731, 744, 760, 775, 1679
 Daten-Insel 1195
 Dateninsel im Dokument 1195
 Datenkapselung 44, 72
 Datenmenge im HTML-Dokument 1195
 Datenquelle als Anker festlegen 560, 567, 597, 609, 648, 666, 681, 686, 691, 700, 703, 708, 713, 725, 730, 743, 759, 774, 1016, 1076, 1605
 Datenquelle-Anzeigart 597, 609, 686, 730, 743, 774
 Datenquelle-Name vergeben 560, 567, 597, 609, 614, 648, 666, 681, 686, 691, 700, 703, 708, 713, 725, 730, 743, 759, 774, 1076, 1603

Datenquelle-Satznummer 562, 598, 610, 649, 667, 682, 692, 700, 709, 713, 726, 731, 744, 760, 775, 1078, 1679
 Datensätze als Teil einer Javascript Datei 46
 Datenspeicherung indiziert 1427
 Datenspeicherung mit Schlüssel 1427
 Daten-Stream 908, 914, 921, 926, 945, 970, 1617
 Datentransfer 1108
 Datentyp 63
 Datentyp array 54
 Datentyp boolean 56, 175
 Datentyp date 56, 176
 Datentyp des Dokumentes 438
 Datentyp einer Variable 63
 Datentyp einer Variablen 63
 Datentyp eines Zeigers 61
 Datentyp ermitteln 54
 Datentyp frei definierbar 62
 Datentyp function 56
 Datentyp Konvertierung 54
 Datentyp Literal 56
 Datentyp number 57
 Datentyp numerisch 57
 Datentyp Objekttyp 58
 Datentyp string 57
 Datentyp und Wert 78
 Datentyp und Zeiger 61
 Datentyp Zeigertyp 58
 Datentypen 54
 Datentypen Basis 54
 Datentypen privat 54
 Datenübersendung an den Server 622
 Datenübertragung 699
 Datum der Dokumenterstellung 681, 1625
 Datum der letzten Dokumentveränderung 427, 681, 1625
 Datum des letzten Datei-Updates 681, 1626
 Datum und Zeit 176
 Datumsangaben 179
 Datumsanzeige in deutscher Norm 179
 Dauer der Animation des Filters 512, 2001
 Dauer der Timeline 889, 1559
 Dauer der Timeline der Wiederholungen laut autoReverse 1686
 Dauer einer Wiederholung 1690
 Dauer Objektaktivitäten 609, 885, 893, 897, 904, 908, 914, 918, 921, 926, 931, 936, 940, 945, 949, 954, 971, 1619
 Dauer Timeline 867
 Dauerdruck auf Taste 1105
 days 748, 1659
 DbClick 1158, 1159
 de 748
 deaktivieren 1095
 deaktivieren HTML-Element 1095
 deaktivierter Text im Dialogfenster 981
 Deaktivierung Objekt 1114, 1116, 1979, 1982
 Debugger 130, 1100, 1147
 Deckungskraft Filter 512, 514, 517, 2002, 2003, 2006
 decodeURI() 149, 159, 1912
 decodeURIComponent() 149, 159, 1912
 default 571, 1598
 defaultChecked 633, 634
 Default-Selektionsstatus document.select.option Objekt 765, 1607
 Default-Selektionsstatus Objekt document.select.option 765, 1607



Default-Selektionsstatus Option 765, 1607
 DEFER 240, 258, 484, 564, 574, 592, 600, 603, 612, 617, 621, 630, 661, 669, 684, 689, 694, 698, 701, 706, 711, 715, 720, 724, 728, 733, 741, 746, 762, 768, 777, 827, 1016, 1028, 1031, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1067, 1070, 1073, 1081, 1191, 1194, 1636, 1829
 de-fokussieren 1097
 Deklaration einer Variable 63
 Deklaration Objektvariable 69
 deklarieren Objektvariable 72
 deklarieren Objektvariable mit new 72
 dekodieren von Literal im Unicode-Format 213
 dekodieren von String oder Literal im Unicode-Format 213
 Delete 277, 476
 delete Anweisung 65, 143
 delete Operator 78, 80
 dependent 438
 deprecated 224, 976
 der Pull-Down-Menüs 397, 439
 Der Windows Media Player 7.1 Plugin für Netscape 1460
 description 748
 Description 748
 Desktop 981
 destination device or buffer 854, 1595
 detaillierten Festsetzung des Suchmusters 1183, 1186
 Deutschland 748
 dezimal 57, 980
 Dezimal zu Hexa 421
 Dezimalkomma 980, 1376, 1828, 1911
 Dezimalkomma zu Dezimalpunkt 200, 1852
 Dezimalpunkt 57, 200, 1852
 Dezimalpunkt zu Dezimalkomma 1828, 1911
 DHMTL 223
 DHTML 225
 DHTML-Eigenschaft 239
 DHTML-Eigenschaft hinzufügen 249, 482, 563, 568, 573, 591, 599, 602, 610, 615, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 Dialekt von Javascript 36
 Dialekte von Javascript 17
 Dialog modal 405, 1901
 Dialog nicht-modal 408, 1903
 Dialogbox 278, 477, 816, 826, 1950
 Dialogbox erzeugen 388, 392, 398, 1738, 1753, 1859
 Dialogbox zu Channel-Einstellung 1162, 1737
 Dialogbox zu Sicherheitseinstellungen 1165, 1901
 Dialogbox zu Spracheinstellungen 1165, 1901
 Dialogbox zur Favoritenverwaltung 1162, 1165, 1738, 1901
 Dialog-Box-Druckfenster 398, 1859
 Dialoge modal oder nicht modal 378, 416, 1608
 Dialogfenster 416, 981
 Dialogfenster Breite 381, 1610
 Dialogfenster Focus 408, 1903
 Dialogfenster Höhe 379, 1609
 Dialogfenster Position 380, 1609, 1610
 dialogHeight 406, 408
 dialogHide 406, 408
 dialogLeft 406, 408
 dialogTop 406, 408

dialogWidth 406, 408
 diamond 1704, 1723
 Dictionary Objekt 1427
 Digital 411, 1601
 DigitallySecure 874, 1469
 Digitaluhr 518
 Dimension 984
 Dimension einer Linie 325, 670
 Dimension Objekt 812, 822, 1937
 Direct Animation ActiveX-Control 1382
 Direct Animation im Internet Explorer 1382
 DirectAnimation 812, 817, 822, 1937
 direction 542
 directories 397, 438
 Directory Buttons 397, 439
 Directory lesen 1559, 1656
 direkt vorhergehende Seite Url 428, 1679
 DISABLED 1635
 disableExternalCapture() 361, 1146
 Disallow: 50
 Disallow: / 50
 Disallow: /pfad_angabe/ 50
 Disallow: /pfad_angabe/datei_angabe 50
 discrete 1588
 display 1712
 display:none; 420, 982, 1679
 DIV 771, 810, 818, 820, 1108, 1236, 1377, 1931
 div Objekt 603, 665
 div Objekt Events 1133, 1954
 div Objekt Filter 549
 div Objekt Styles 831
 Division ganzzahlig 198, 203
 DIV-Objekt 224
 do while Anweisung 87
 DOCTYPE 590, 610, 663, 1684
 document 1240
 Document Object Model 239, 1825
 document Objekt 281, 283, 370, 381, 428, 801, 1614, 1714, 1754
 document Objekt des Kotes 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1667
 document Objekt des Netscape 304
 document Objekt Events 1133, 1955
 document Objekt Filter 549
 document Objekt Styles 832
 Document Type Definition-Version 663, 1731
 document.all Collection 447, 453, 456, 482, 598, 602, 628, 648, 649, 651, 660, 663, 665, 666, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 736, 740, 745, 750, 754, 757, 763, 769, 1000, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1188, 1597, 1692
 document.all Index des Objektes 610, 775
 document.anchors Collection 448, 565
 document.applets Collection 450, 570
 document.body Objekt 584, 866
 document.body.timeAll Collection 451, 593, 865
 document.captureEvents(Event.XXX); 1143
 document.cookie Collection 488, 974
 document.embeds Collection 451, 613, 756, 1170, 1171
 document.form Objekt 622
 document.form.elements Collection 452, 641
 document.form.input Objekt 632, 685



document.form.input.button Objekt 632
 document.form.input.checkbox Objekt 633
 document.form.input.fileupload Objekt 634
 document.form.input.hidden Objekt 634
 document.form.input.password Objekt 635
 document.form.input.radio Objekt 635
 document.form.input.reset Objekt 637
 document.form.input.submit Objekt 638
 document.form.input.text Objekt 638
 document.forms Collection 452, 641
 document.frames Collection 382, 453, 648, 651, 658, 665, 1627, 1651
 document.html Objekt 662
 document.images Collection 454, 671, 684, 707
 document.links Collection 455, 738
 document.location.reload() 222
 document.namespace Objekt 504, 851, 852, 855, 856, 857, 862, 863, 864
 document.namespaces Collection 504, 506, 507
 document.open() 1104
 document.scripts Collection 1190
 document.select Objekt 757, 759, 760, 761, 763, 765, 766, 767, 1607, 1635, 1649, 1658, 1666, 1687, 1692, 1711, 1723
 document.select Objekt Listbox 766, 1635
 document.select.option Objekt 763, 1666
 document.select.option Objekt Default-Selektionsstatus 765, 1607
 document.select.option Objekt interner Wert 767, 1711
 document.select.option Objekt Label 766, 1649
 document.select.option Objekt laufende Nummer in List-Box 765, 1635
 document.select.option Objekt Selektionsstatus 766, 1687
 document.select.options Collection 456, 757, 760, 763, 769, 1666, 1687
 document.selection Objekt 456, 457, 458, 770, 771, 1083, 1723, 1746, 1757, 1764, 1883
 document.selection Objekt löschen Selektion 770, 1764
 document.selection Objekt Markierung der Selektion 770, 1746
 document.selection Objekt Selektion löschen 770, 1764
 document.selection Objekt Selektion Markierung 770, 1746
 document.selection Objekt Selektion Typ 770, 1723
 document.selection Objekt Selektion Universal-Bereich erzeugen 458, 770, 1083, 1757
 document.selection Objekt Typ der Selektion 770, 1723
 document.selection Objekt Universal-Bereich erzeugen Selektion 458, 770, 1083, 1757
 document.selection.controlrange Collection 456, 770
 document.selection.controlRange Collection 458, 770, 1083, 1757, 1764
 document.selection.textrange Collection 457, 458, 770, 771, 1083, 1757, 1764
 document.styleSheets Collection 994, 999
 document.title 382, 1659
 document.write() 43
 document.write() und Rekursion im HEAD 113
 document.writeln() 43
 document.writeln() und Rekursion im HEAD 113
 document-Events 1209, 1965
 document-Objekt 239
 Dokument aktive Selektion 770
 Dokument aktiver Link 1564
 Dokument aktuelles 1165

Dokument Alink 1564
 Dokument alle beinhaltete Objekte 256, 435, 629, 827, 1786
 Dokument als Datei speichern 1129, 1996
 Dokument Animation von Elementen 508
 Dokument Anzeige beenden 1749
 Dokument Attribut erzeugen 251, 431, 826, 1753
 Dokument Attribute Wirksamkeit 426
 Dokument ausdrucken 584
 Dokument automatisch scrollen 1876
 Dokument Baumhierarchie 222
 Dokument BODY-Abschnitt 40
 Dokument Bottom Margin 1582
 Dokument Charset 626, 1556
 Dokument Cookie 424, 1597
 Dokument Daten ausgeben 447, 1913
 Dokument Daten-Insel 1195
 Dokument Datenmenge 1195
 Dokument Domain-Suffix 426, 1616
 Dokument drucken 398, 419, 1859
 Dokument dynamischen Eigenschaften neu berechnen 827, 1860
 Dokument dynamischen Eigenschaftenveränderung zur Laufzeit 807
 Dokument Ebenendarstellung 772
 Dokument Editierbarkeit 425, 1607
 Dokument Eigenes Dokument wird durch fremde Webseite geladen 313, 320, 643, 653
 Dokument Element animieren 508
 Dokument Elternfenster 428, 1670
 Dokument entladen 1114, 1979
 Dokument erzeugen 432, 1754
 Dokument Event-Objekt erzeugen 433, 1107, 1755
 Dokument Farbe bereits geklickter Links 1731
 Dokument Farbe eine VLINK 1731
 Dokument Farbe eines Links 1651
 Dokument Farbe von noch nicht benutzten Links 427, 1651
 Dokument Feld aller Anker 448, 565
 Dokument Feld aller Applet-Objekte 450, 570
 Dokument Feld aller Elemente 447
 Dokument Feld aller Formulare 452, 641
 Dokument Feld aller Frames 453, 651
 Dokument Feld aller img Objekte 326, 454, 684, 707
 Dokument Fenster schliessen 391, 1750
 Dokument Fenstertitel 428, 1714
 Dokument Filter 508, 509
 Dokument fremdes Dokument laden ohne Framedarstellung 313, 319, 643, 653
 Dokument für Suchmaschine vorbereiten 748
 Dokument Fusszeile 419
 Dokument Größe 427, 681, 1625
 Dokument HEAD-Abschnitt 40
 Dokument Hintergrundbild 1574
 Dokument Hintergrundbild Scrollbarkeit 1581
 Dokument Hintergrundfarbe 424, 589, 743, 1581
 Dokument History-Eintrag ersetzen 1167, 1869
 Dokument HTML-Schnappschuss 864
 Dokument im Fenster scrollen 400, 401, 1876
 Dokument im Fenster verschieben 400, 401, 1876
 Dokument in das Fenster laden 396
 Dokument instanzieren 437, 1849
 Dokument Kompatibilität des IE 6.x zu CSS1 424, 1595
 Dokument Kopfzeile 419
 Dokument Lade-Ereignisse 1162
 Dokument laden 222, 395, 1165, 1167, 1739, 1844, 1869



Dokument Laden eines fremden Dokumentes ohne Framedarstellung 313, 319, 643, 653
 Dokument Layout 419
 Dokument Layout auf einen Schlag ändern 994
 Dokument linke obere Ecke 1104
 Dokument löschen 431, 1746
 Dokument mehrere bgsound Objekte 576
 Dokument mehrsprachige Stichwortliste 748
 Dokument Mime-Typ 437
 Dokument mit embedded Objekte laden 369, 1168
 Dokument mit Plugins laden 369, 1168
 Dokument nachliegende 1165
 Dokument neu laden 278, 477, 1103, 1167, 1865
 Dokument neues erzeugen 432, 1754
 Dokument neuladen 222
 dokument Objekt Kompatibilität des IE 6.x zu CSS1 424, 1595
 Dokument Objekt styleSheet erzeugen 433
 Dokument Objekthierarchie 69
 Dokument öffnen 437, 1849
 Dokument ohne linken und oberen Standard-Seitenrand 584
 Dokument Plain-Textelement 253, 280, 1757
 Dokument Refresh 369, 1168
 Dokument refreshen 278, 477
 Dokument Reload 772
 Dokument Reload mit allen seinen FRAMES 315, 321, 644, 655
 Dokument Rumpf 584
 Dokument schliessen 431, 1749
 Dokument Script 1190
 Dokument Seitenrand 584
 Dokument speichern 278, 477
 Dokument speichern auf Festplatte 864
 Dokument Standard-Seitenrand 584
 Dokument Stichwortliste mehrsprachig 748
 Dokument Styles auf einen Schlag ändern 994
 Dokument styleSheet Objekt erzeugen 433
 Dokument Style-Sheet-Objekt 253, 826, 1757
 Dokument Suche von Text 1162
 Dokument Textfarbe 1625
 Dokument Text-Objekt erzeugen 433
 Dokument Timeline 866
 Dokument Top Margin 1721
 Dokument und sein Textbereich 458, 1082
 Dokument und Selektion 458, 770, 1083, 1723, 1724, 1746, 1757, 1764
 Dokument und Style-Sheet (CSS) 976
 Dokument Uniform Resource Name (URN) 507, 563, 1726
 Dokument Unload 1114, 1131, 1979, 1999
 Dokument Url 396, 430, 1726
 Dokument Url auf Server 626, 1558
 Dokument Userdaten permanent sichern 976, 1624, 1837, 1874
 Dokument Userdaten permanent speichern 976, 1624, 1837, 1874
 Dokument Userdaten verwalten 976, 1624, 1837, 1874
 Dokument verlassen 1104, 1114, 1129, 1979, 1996
 Dokument Verlauf 281, 283, 370, 382, 1633
 Dokument vom Cache auf Festplatte laden 735, 1166
 Dokument vom Server und nicht Cache laden 735, 1166
 Dokument Vordergrundfarbe 426, 1625
 Dokument vorhergehende Seite 1679
 Dokument Wurzelknoten 243, 425, 1616
 Dokument XML 249, 430, 863, 864, 1188, 1195, 1734

Dokument XSL 249, 430, 1734
 Dokument Zeichensatz 424, 560, 736, 749, 1187, 1591
 Dokument zuerst gefundenes Objekt 255, 435, 827, 1786
 Dokument ZUERST gefundenes Objekt 255, 435, 827, 1786
 Dokument zuvorliegendes 1165
 dokumentbezogener Filter 509
 Dokumentdarstellung 1568
 Dokumente Kommunikation 426, 1616
 Dokumenten-Hintergrund auswählen 422
 Dokumenterstellung Datum 681, 1625
 Dokumentfenster 981
 Dokument-Layout 222
 Dokumenttyp 425, 1614
 Dokumentveränderung letzte Datum 427, 681, 1625
 DOM 261, 504, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 741, 747, 752, 756, 763, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1167, 1194, 1868
 DOM attributes Collection 266, 268, 275, 464, 468, 474
 DOM Collection attributes 266, 268, 275, 464, 468, 474
 DOM Elementeigenschaft 250, 482, 563, 568, 573, 591, 599, 602, 610, 616, 620, 629, 649, 660, 663, 668, 682, 688, 693, 697, 701, 705, 710, 714, 719, 723, 727, 732, 737, 740, 745, 754, 761, 767, 776, 1019, 1029, 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1739
 DOM entfernen eines Objektes 223
 DOM entfernen von Kind 223
 DOM HTML 239, 1825
 DOM Inkonsistenz 239
 DOM Kind entfernen 223
 DOM Konsistenz 222, 239
 DOM normalisieren 239
 DOM Normalisierung 259, 484, 564, 569, 574, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 762, 768, 777, 1025, 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1196, 1848
 DOM Objekt entfernen 223
 DOM Objektzugehörigkeit zum DOM 239, 1825
 DOM und HTML-Dokument 418
 DOM und HTML-Element 418
 DOM XML 239, 863, 864, 1188, 1825
 Domain 561, 572, 1167, 1633
 Domain einer Homepage 857, 1830
 Domain Homepage 856
 domain= 488
 Domain-Suffix des Dokumentes 426, 1616
 DOM-Hierarchie 222
 DOM-Position Knoten 240
 Doppelklick 1098
 Doppelpunkt 977, 984
 doScroll() 483, 591, 611, 777, 1079, 1763
 dotted 983, 987, 991
 double 983, 987, 991
 down 483, 1080, 1610, 1763
 Download Datei 559
 Download einer Datei 855
 Download Script 1188, 1607
 Download Start 856, 1906
 Download vom Image 1112, 1977
 Drag 1108
 Drag & Drop 1098, 1099, 1104, 1108, 1116, 1150, 1983



Drag & Drop beim Internet Explorer 1150
 Drag & Drop beim Netscape 1152
 Drag & Drop Eventarten 1150
 Drag und Drop 1108
 Drag und Drop Art 1112, 1620
 dragdrop 1152
 Dragging 277, 476
 Dragging-Eigenschaft 1620
 Drag-Manipulation Status 563, 574, 591, 599, 602, 611, 620, 629, 650, 664, 668, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 740, 746, 750, 755, 761, 767, 777, 1019, 1030, 1034, 1060, 1066, 1080, 1189, 1763
 Drop 1108, 1112, 1617
 Drop-Down-Liste 757
 Drop-Down-Selections-Control 277, 477
 DropShadow Filter 526
 Druck 1114, 1979
 Druck des Dokumentes 419
 Druck eines Fensters oder Frames 1097
 Druck eines Frames 1097
 Druck Ende 1112, 1977
 Druckbutton 398, 1859
 Druck-Dialogbox 278, 477
 drücken irgendeiner Maustaste 1101
 drücken mittlere Maustaste 1101
 Druck-Eventbehandlung nur Internet Explorer ab 5.x 1161
 Druckfenster 398, 1859
 Druckvorschau 1114, 1120, 1655, 1979, 1986
 Druckvorschau Ende 1112, 1977
 DSO 569, 753, 755, 1195, 1196, 1679, 1842
 DTD-Version 663, 1731
 dünn 987, 989
 duplizieren Textbereich 459, 1084, 1764
 DUR 867
 Duration 749
 durchgezogener Trennstrich 987
 durchreichen Event 401, 446, 484, 564, 570, 575, 593, 601, 612, 618, 621, 631, 651, 662, 669, 684, 690, 694, 699, 702, 707, 711, 716, 720, 725, 729, 734, 747, 756, 763, 778, 1027, 1031, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1194, 1884, 1886
 DXImageTransform.Microsoft 509
 DXImageTransform.Microsoft.Alpha 509
 DXImageTransform.Microsoft.Barn 510
 DXImageTransform.Microsoft.BasicImage 509
 DXImageTransform.Microsoft.Blinds 510
 DXImageTransform.Microsoft.CheckerBoard 510
 DXImageTransform.Microsoft.Chroma 509
 DXImageTransform.Microsoft.DropShadow 509
 DXImageTransform.Microsoft.Fade 509
 DXImageTransform.Microsoft.Glow 509
 DXImageTransform.Microsoft.Iris 510
 DXImageTransform.Microsoft.Light 510
 DXImageTransform.Microsoft.MaskFilter 510
 DXImageTransform.Microsoft.MotionBlur 509
 DXImageTransform.Microsoft.RandomBars 510
 DXImageTransform.Microsoft.RandomDissolve 510
 DXImageTransform.Microsoft.Shadow 510
 DXImageTransform.Microsoft.Strips 510
 DXImageTransform.Microsoft.Wave 510
 dynamische HTML-Programmierung 225
 dynamische Zeigerverwaltung 58
 dynamischen Eigenschaften des Dokumentes neu berechnen 827, 1860

Dynamischen Eigenschaftenveränderung zur Laufzeit 807
 dynamischer Filter 508
 dynamischer Scriptcode 211
 dynamisches Feld 43, 44, 71
 dynamisches Objektzeigerfeld 71
 dynamisches Zeigerfeld 43
 DYNsrc 671
 dzeichenfolge 1184
 Dzeichenfolge 1184
 E 199, 1620
 Ebenendarstellung 772
 Ecke des HTML-Dokumentes 1104
 ECMA-Script 1723
 edge 406, 408
 Editierbarkeit Content vom Objekt 481, 560, 589, 597, 609, 614, 618, 627, 686, 708, 713, 717, 721, 725, 730, 743, 774, 1076, 1191, 1596
 Editierbarkeit des Dokumentes 425, 1607
 Editierbarkeit des Objekt-Content 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1640
 Editierbarkeit eines Text-Controls 1679
 Eigenes Dokument wird durch fremde Webseite geladen 313, 320, 643, 653
 Eigenschaft Art 1107, 1676
 Eigenschaft des attribute-Objektes 255, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 751, 755, 761, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1196, 1778
 Eigenschaft Drag & Drop 1620
 Eigenschaft eines JScript-Objektes 155, 211, 1825
 Eigenschaft Element 260, 484, 564, 569, 575, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 738, 741, 747, 751, 755, 762, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1196, 1867
 Eigenschaft hinzufügen 249, 482, 563, 568, 573, 591, 599, 602, 610, 615, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 Eigenschaft implementiert 223
 Eigenschaft Kind oder Eltern 240
 Eigenschaft Name 1107, 1676
 Eigenschaft Objekt ändern 1126, 1993
 eigenschaft: wert; 984
 eigenschaft=wert; 984
 Eigenschaften eines Filters 511
 Eigenschaften IE Standard 249, 482, 563, 568, 573, 591, 599, 602, 610, 616, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1019, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 Eigenschaften in einem Berechnungsausdruck 827, 1860
 Eigenschaften und Methoden 111
 Eigenschaften vom Cookie 488
 Eigenschaften zu <A> 984



Eigenschaften zu <H1> bis <H6> 984
 Eigenschaften zu <P> 984
 Einblende eines Bildes 528, 671
 Einblende von Bildern mit Bildwechsel 528, 672
 Eine bestimmte Suchmaschinen darf NICHT scannen 49
 Eine bestimmte Suchmaschinen darf scannen 49
 Eingabe in Adresszeile 428, 1679
 Eingabedaten für Server 626, 1556
 Eingabefeld aktivieren 634
 Eingabefeld auslesen 639
 Eingabefeld deaktivieren 634
 Eingabefeld Wert zuweisen 640
 Eingabefenster 398, 1859
 Eingabezeile 398, 1859
 Eingabezeile Adresse 397, 439
 Eingabezeile Url 397, 439
 einlesen der Werte der Argumente aus dem
 Funktionsaufruf 104, 161, 162, 1553, 1650
 einmaliger Aufruf eines Scripts per Timer 404, 1895
 Einrichten eines virtuellen Hosts per Apache-HTTP-
 Server 24
 Einrichtung einer Webseite per Apache-HTTP-Server 25
 einrücken 986
 Einstellungen System-lokal 158, 212, 1910
 Einstellungen zu Sicherheit 1162
 einzeilige Textbox 725
 Einzug 980
 Element animieren 907
 Element Attribute 259, 437, 484, 564, 569, 574, 592,
 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684,
 689, 694, 698, 702, 706, 711, 715, 720, 724, 728, 733,
 738, 741, 747, 755, 762, 768, 777, 827, 1023, 1031,
 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081,
 1189, 1194, 1838
 Element Behavior 809, 819, 1919
 Element Beschriftung 729
 Element Daten Drag und Drop 1108
 Element den Focus wegnehmen 483, 563, 568, 573, 591,
 599, 611, 616, 620, 629, 649, 660, 668, 683, 688, 693,
 697, 705, 710, 714, 719, 723, 727, 732, 745, 754, 761,
 776, 1019, 1030, 1043, 1048, 1053, 1060, 1066, 1072,
 1079, 1193, 1745
 Element Drag und Drop von Daten 1108
 Element Eigenschaft 260, 484, 564, 569, 575, 592, 600,
 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689,
 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 738,
 741, 747, 751, 755, 762, 768, 778, 1027, 1031, 1035,
 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1190,
 1194, 1196, 1867
 Element eines Formulars 686, 691, 696, 700, 703, 708,
 713, 717, 721, 725, 1607
 Element entfernen aus einer Collection 456, 457, 742,
 750, 762, 770, 771, 1865
 Element erzeugen 222, 587
 Element Erzeugung durch Makro 275, 474
 Element getimt 1713
 Element in einem Formular 597, 614, 633, 634, 635,
 636, 637, 638, 641, 686, 691, 696, 700, 703, 708, 713,
 717, 721, 726, 730, 753, 759, 765, 1077, 1627
 Element innerhalb eines Elementes 251, 483, 563, 569,
 574, 591, 599, 611, 616, 620, 629, 650, 660, 664, 668,
 683, 688, 693, 698, 705, 710, 715, 719, 723, 728, 732,
 737, 740, 746, 750, 761, 767, 776, 1019, 1030, 1034,
 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189,
 1193, 1753

element Objekt 490, 499, 501, 502, 503, 504, 851, 1147,
 1755, 1770, 1772, 1981, 1983, 1993
 Element scrollen 1103
 Element Verhalten 809, 819, 1919
 Element Verhaltensweise 490, 504
 Element Zugriffe per Tastaturkürzel 729
 Element Zustand in der Timeline 866
 Element-Ausschnitt 985
 Elemente im DOM 223
 Elemente mit gemeinsamer URN 450, 451, 452, 566,
 571, 614, 757
 Elemente verschachtelt 1103
 Elementeneigenschaft im DOM 250, 482, 563, 568, 573,
 591, 599, 602, 610, 616, 620, 629, 649, 660, 663, 668,
 682, 688, 693, 697, 701, 705, 710, 714, 719, 723, 727,
 732, 737, 740, 745, 754, 761, 767, 776, 1019, 1029,
 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079,
 1189, 1192, 1739
 Elementeneigenschaft Kind oder Eltern 240
 Elementes Tab-Tasten-Folge 482, 562, 568, 573, 598,
 610, 619, 628, 649, 660, 667, 682, 687, 692, 697, 704,
 709, 714, 718, 722, 727, 731, 745, 754, 776, 1018,
 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192,
 1708
 Element-Knoten 238, 244, 1017, 1028, 1032, 1036,
 1042, 1047, 1052, 1058, 1064, 1070, 1661
 Element-Lade-Eventbehandlung 1161
 Element-Scrolling 985
 Element-Sichtbarkeit im Dokument 222
 ellipseWipe 1704, 1723
 Eltern 222, 250, 482, 563, 568, 573, 591, 599, 602, 610,
 616, 620, 629, 649, 660, 663, 668, 682, 688, 693, 697,
 701, 705, 710, 714, 719, 723, 727, 732, 737, 740, 745,
 754, 761, 767, 776, 1019, 1029, 1034, 1037, 1043,
 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1739
 Eltern Body 482, 561, 568, 572, 590, 598, 601, 615,
 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700,
 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750,
 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037,
 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192,
 1195, 1667
 Eltern Referenz 482, 561, 567, 572, 598, 601, 610, 619,
 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713,
 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028,
 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077,
 1191, 1663
 Elterneigenschaft 240
 Elternelement des Textbereiches 460, 1084, 1850
 Elternfenster 283, 1709
 Elternfenster als Erzeuger eines Fensters 221, 382, 1666
 Elternfenster des Dokumentes 428, 1670
 Elternframe 1709
 Elternknoten 240, 245, 561, 568, 572, 590, 598, 601,
 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696,
 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744,
 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042,
 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668
 Elternobjekt 240
 Elternobjekt Abstand zum Kind 666, 681, 1634
 Elternobjekt Textbereich 240, 245, 482, 562, 568, 572,
 590, 598, 602, 615, 619, 649, 659, 663, 667, 682, 687,
 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736,
 740, 744, 750, 753, 760, 766, 775, 1017, 1029, 1033,
 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1077, 1188,
 1192, 1668
 Elternordner 1433, 1440, 1668, 1800



EM 815, 824, 1945
 E-Mail 622, 638, 1097
 EMBED 369, 1168
 embed Objekt Events 1134, 1955
 embed Objekt Styles 832
 Emboss Filter 526
 EMF 671, 702
 Empfangen der Daten vom Server 627, 1656
 en 748
 enablePlugin 1169, 1170
 Encoden der Eingabedaten durch den Server 626, 1556
 encodeURI() 144, 153, 1766
 encodeURIComponent() 144, 153, 1766
 ENCTYPE 622, 695
 END 867
 End Of Line 1443, 1568
 End Of Stream 1443, 1568
 Ende der Timeline 867
 Ende des Druck 1112, 1977
 Ende Druckvorschau 1112, 1977
 Ende-Tag 222
 Endezeit als Offset von der Startzeit der Eltern-Timeline 1669
 endif 38, 39
 endlos scrollen 1653
 endlos scrollen marquee Objekt 1653
 endlos scrollen Objekt marquee 1653
 Englisch 748
 Engrave Filter 526
 Enter 1101
 entfernt Cursor vom Eingabefeld 634, 635
 entfernt Cursor von der Checkbox 634
 entfernt den Cursor aus dem Eingabefeld 641
 entfernt den Cursor vom Resetbutton 637
 entfernt den Cursor vom Submitbutton 638
 entladen eines Dokumentes 1114, 1979
 Entwertung Backslash 56, 58, 114
 Entwertung des Zeichens ' 56, 58, 114
 Entwertung des Zeichens " 56, 58, 114
 Enumerator JScript-Objekt 149, 188, 1740, 1831, 1840
 Enumerator Objekt 149, 160
 Enumerator Script-Objekt 160
 EOL 1443, 1568
 EOS 1443, 1568
 Ereignis als Objekt 1143
 Ereignis als Parameter 1143
 Ereignis Standardbehandlung 433, 483, 507, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1107, 1189, 1193, 1759
 Ereignis und Eventhandler 1143
 Ereignis weiterreichen 1098
 Ereignisbehandlung für mouseover und mouseout ein- bzw. ausschalten 1147
 Ereignis-Handler überschreiben 291
 Ereignishandler und Script 1188, 1634
 Ereigniskontrolle 361, 1145
 Ereignisprogrammierung 1100
 Ereignisse für Bild 1161
 Ereignisse für Bildladen 1161
 Ereignisse Objekt window 291, 298
 Ereignisüberwachung in einem Fenster 293
 Ereignisüberwachung per Rekursion 401, 404, 1892, 1895

e-resize 988
 E-resize 993
 Erkennung der Javascript-Version 39
 Ermittlung der Objektklasse einer Objektinstanz 142
 error JScript-Objekt 192, 1607, 1656, 1659, 1662
 Error JScript-Objekt 149
 Error Objekt 149, 160
 Error Script-Objekt 160
 Ersatz von Cookies durch input hidden Objekt 699
 ersetzen History-Eintrag 397, 440
 ERSTES Kind 241
 erstes Kind Referenz 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 erstes Kind Zeiger 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 Erweiterung Objektvariable 70
 Erweiterungen browserhersteller-spezifisch 223
 Erzeugbarkeit eines Textbereiches 627, 1647
 erzeugen Attribut automatisch und mit Wert belegen 262, 484, 565, 570, 575, 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 erzeugen Attribut und mit Wert belegen 262, 484, 565, 570, 575, 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 erzeugen eines Dokumentes 432, 1754
 erzeugen eines Url-Wertes 337, 780, 828
 erzeugen Fenster 437, 1849
 erzeugen Objekt window 437, 1849
 Erzeuger des Fensters 283
 Erzeugung eines 2-dimensionalen Feldes 77, 163
 Erzeugung eines Ordners 1450
 Escape Sequenzen 56, 58, 114
 escape() 144, 317, 323, 646, 657
 Escape-Sequenzen 152, 153, 1764
 ESC-Sequenz 113
 etwas größer als normal 987, 989
 etwas kleiner als normal 987, 989
 Eurozeichen 113
 event 1242
 EVENT 41, 1095
 Event Art 1107, 1721
 Event ausgelöst durch Maustaste 1107, 1584
 Event auslösen 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 997, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1107, 1189, 1193, 1196, 1772
 Event auslösendes Objekt 1107, 1695
 Event behandeln 1147
 Event bei Behavior 1147
 Event Bezeichner 1107, 1721



Event des IE einer Nicht-Standardbehandlung unterziehen 1147

Event des Netscape einer Nicht-Standardbehandlung unterziehen 1144

Event des Netscape von Nicht-Standardbehandlung wieder der Standardbehandlung unterziehen 1145

Event durchreichen 401, 446, 484, 564, 570, 575, 593, 601, 612, 618, 621, 631, 651, 662, 669, 684, 690, 694, 699, 702, 707, 711, 716, 720, 725, 729, 734, 747, 756, 763, 778, 1027, 1031, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1194, 1884, 1886

Event durchreichen über Eventhandlerkette 1107, 1589

Event eines Behavior per *.htc-Datei 1147

Event entlang der Eventhierarchie weiterreichen 1144

Event entlang der Nicht-Standard- Eventhierarchie weiterreichen 1145

Event entlang der Standard-Eventhierarchie weiterreichen 1145

Event im Standard-Behavior 1147

Event Kodierung im HTML-Tag 354, 1093

Event Lade-Ereignisse beim Netscape 365

Event Maus auslösendes Objekt 1107, 1720

Event Mouse-Eventbehandlung beim Netscape 365

event Objekt 41, 281, 283, 370, 381, 1087, 1095, 1623, 1772

event Objekt Styles 832

event Objekt Url laut einem Kommando aus einer Advanced Streaming Format (ASF) Datei 917, 930, 1107, 1131, 1726, 2000

Event onblur 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 649, 660, 668, 683, 688, 693, 697, 705, 710, 714, 719, 723, 727, 732, 745, 754, 761, 776, 1019, 1030, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1193, 1745

Event ondragstart 1763

Event onfocus 435, 483, 563, 569, 574, 599, 611, 620, 629, 650, 661, 668, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 746, 755, 777, 1021, 1030, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1193, 1775

Event onmouseover Pixelgenauigkeit 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751

Event registrieren 388, 430, 482, 507, 563, 568, 573, 591, 599, 602, 610, 616, 620, 629, 649, 660, 663, 668, 683, 688, 693, 697, 701, 705, 710, 714, 719, 723, 727, 732, 737, 740, 745, 754, 761, 767, 776, 1019, 1029, 1034, 1038, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1107, 1189, 1193, 1740, 1741

Event Tastatur-Eventbehandlung beim Netscape 363

Event und Eventhandler dem Objekt window zuordnen 1144

Event und Script 1188, 1623, 1634

Event von Nicht-Standardbehandlung wieder der Standardbehandlung unterziehen 1147

Event weiterreichen 300

Event.BLUR 291, 298, 359, 360, 1144, 1145

event.bookmarks Collection 1108

event.dataTransfer Objekt 378, 413, 1108, 1593, 1617, 1620

Event.DRAGDROP 291, 298, 359, 360, 1144, 1145

Event.ERROR 291, 298, 359, 360, 1144, 1145

Event.FOCUS 291, 298, 359, 360, 1144, 1145

Event.KEYDOWN 359, 360, 1144, 1145

Event.KEYPRESS 359, 360, 1144, 1145

Event.KEYUP 359, 360, 363, 1144, 1145

Event.LOAD 292, 359, 360, 1144, 1145

Event.MOUSEDOWN 300, 359, 360, 361, 1144, 1145, 1146

Event.MOUSEOVER 292, 298, 359, 360, 1144, 1145

Event.MOVE 292, 298, 359, 360, 1144, 1145

Event.RESIZE 292, 298, 359, 360, 1144, 1145

event.returnValue = true; 1143

event.returnValue=true; 1105

event.srcElement.id 1103

event.srcElement.name 1103

Event.UNLOAD 292, 298, 359, 360, 1144, 1145

Eventart 1104

Event-Art 1106

Eventarten 1095

Eventarten der HTML-Tags 1095

Eventarten des IE und NS 1097

Eventarten für Drag & Drop 1150

Eventarten wichtiger Objekte 1131

Event-Behandlung 1142

Eventbehandlung bei Drag & Drop 1150

Eventbehandlung beim Internet Explorer 1147

Eventbehandlung des Netscape 359

Eventbehandlung des Netscape Beispiele 362

Eventbehandlung des Netscape für eine Seite mit Frame 361, 1145

Eventbehandlung durch eine Fremdseite mit signiertem Script 1145

Eventbehandlung durch Fremde Seite aktivieren 1146

Eventbehandlung durch Fremde Seite deaktivieren 1146

Eventbehandlung für Textoperationen mit der Windows-Zwischenablage 1160

Eventbehandlung im Netscape 1144

Eventbehandlung in einem Formular 1149

Eventbubbling 1142

Eventcapturing 1142

Eventdurchreichung aktivieren 446, 484, 564, 570, 575, 593, 601, 612, 618, 621, 631, 651, 662, 669, 684, 690, 694, 699, 702, 707, 711, 716, 720, 725, 729, 734, 747, 756, 763, 778, 1027, 1031, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1194, 1884

Event-Eigenschaften prüfen 1143

Eventhandler 300, 391, 395, 1143, 1750

Eventhandler aufrufen 294

Eventhandler für Tastatur- und Mausereignisse 1143

Eventhandler Parameter 1143, 1147

Eventhandler retten 388, 393, 1741, 1759

Eventhandler Returnwert 1107, 1681

Eventhandler Rückkehrcode 1105

Eventhandler Rückkehrcode 1159

Eventhandler und Ereignis 1143

Eventhandlerkette 1107, 1589

Eventhierarchie 300, 360, 1145

EventLOAD 298

Eventname 1721

Eventobjekt des Netscape 330

Eventobjekt für Maus 1107, 1630

Event-Objekt im Dokument erzeugen 433, 1107, 1755

Eventobjekt und Tastatur 1107, 1647

Events 238

Events a Objekt 1132, 1953

Events applet Objekt 1132, 1953

Events area Objekt 1132, 1953

Events bgsound Objekt 1132, 1954

Events body Objekt 1132, 1954



Events button Objekt 1133, 1954
 Events comment Objekt 1133, 1954
 Events custom Objekt 1133, 1954
 Events div Objekt 1133, 1954
 Events document Objekt 1133, 1955
 Events embed Objekt 1134, 1955
 Events fieldset Objekt 1134, 1955
 Events font Objekt 1134, 1955
 Events form Objekt 1134, 1956
 Events frame Objekt 1135, 1956
 Events frameset Objekt 1135, 1956
 Events head Objekt 1135, 1956
 Events html comment Objekt 1135, 1956
 Events html Objekt 1135, 1956
 Events iframe Objekt 1135, 1956
 Events img Objekt 1135, 1956
 Events input button Objekt 1135, 1957
 Events input checkbox Objekt 1136, 1957
 Events input file Objekt 1136, 1957
 Events input hidden Objekt 1136, 1957
 Events input image Objekt 1136, 1957
 Events input Objekt 1135, 1957
 Events input password Objekt 1136, 1958
 Events input radio Objekt 1137, 1958
 Events input reset Objekt 1137, 1958
 Events input submit Objekt 1137, 1958
 Events input text Objekt 1137, 1959
 Events label Objekt 1138, 1959
 Events link Objekt 1138, 1959
 Events map Objekt 1138, 1959
 Events marquee Objekt 1138, 1959
 Events namespace Objekt 1139, 1960
 Events noframe Objekt 1139, 1960
 Events noscript Objekt 1139, 1960
 Events object Objekt 1139, 1960
 Events option Objekt 1139, 1960
 Events popup Objekt 1139, 1960
 Events registrieren 433, 483, 507, 563, 569, 574, 591,
 599, 602, 611, 616, 620, 629, 650, 660, 664, 668, 683,
 688, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732,
 737, 740, 746, 755, 761, 767, 777, 1019, 1030, 1034,
 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1107,
 1189, 1193, 1759
 Events registrieren 393, 1759
 Events script Objekt 1139, 1960
 Events select Objekt 1139, 1960
 Events selection Objekt 1139, 1960
 Events span Objekt 1139, 1960
 Events Standardbehandlung 300
 Events Standardhierarchie 294
 Events style Objekt 1140, 1961
 Events table Objekt 1140, 1961
 Events table.caption Objekt 1140, 1961
 Events table.col Objekt 1140, 1961
 Events table.colGroup Objekt 1140, 1961
 Events table.tBody Objekt 1140, 1961
 Events table.tFoot Objekt 1140, 1962
 Events table.tHead Objekt 1141, 1962
 Events table.tr Objekt 1141, 1962
 Events table.tr.td Objekt 1141, 1962
 Events table.tr.th Objekt 1141, 1963
 Events textarea Objekt 1142, 1963
 Events textrange Objekt 1142, 1963
 Events var Objekt 1142, 1963
 Events window Objekt 1142, 1963
 Events xml Objekt 1142, 1963

Eventshierarchie 294
 Eventtyp 1772
 Event-Typ 1106, 1153, 1156, 1159, 1160
 Eventüberwachung per Rekursion 401, 404, 1892, 1895
 Eventverarbeitungshierarchie 294
 excl 1713
 EXCL 869
 exklusiv 868
 Existenz einer Variablen prüfen 64
 Existenz Kind möglich 242, 481, 560, 597, 601, 614,
 618, 627, 659, 662, 666, 686, 691, 695, 703, 708, 712,
 717, 721, 725, 730, 739, 743, 752, 759, 765, 774,
 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063,
 1069, 1076, 1191, 1589
 Existenz von Kinder 258, 564, 569, 574, 592, 600, 602,
 611, 617, 621, 630, 650, 661, 664, 668, 683, 689, 694,
 698, 706, 710, 715, 719, 724, 728, 733, 737, 741, 746,
 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049,
 1054, 1061, 1067, 1073, 1081, 1189, 1193, 1824
 Existenz von Kindern 240
 Expires 748, 1596, 1635
 expires= 489
 Exponential-Darstellung 205, 1909
 export 101
 Expression 102
 externes Objekt 70
 extra dünn 987, 989
 extra fett 987, 988, 989
 Extra-Internet Optionen-Sicherheit 665
 Extras-Internetoptionen-Inhalte 1572
 F1 Hilfe 1119, 1986
 F1-Hilfe 1100
 F1-Taste 1100
 fade 1704, 1723
 Fade Filter 527
 Fadenkreuz 988
 FAVORITEN 1108
 false 56, 57, 75, 76
 Familie des Font-Typs 618, 1625
 fanWipe 1704, 1723
 Farbangaben 980
 Farbanteil dunkel für Schatten bei 3D-Element 981
 Farbanteil dunkel für Schatten bei 3D-Element 981
 Farbanteil hell für Schatten bei 3D-Element 981
 Farbbereiche 981
 Farbbits pro Bildpunkt 1182
 Farbe 3D-Schatten von Button im Dialogfenster 981
 Farbe aktiven Fenstertitelzeile 981
 Farbe aktives Fenster 981
 Farbe aller aktiven Links 424, 1564
 Farbe Anzahl der Bits pro Pixel 854, 1583
 Farbe ausgewählter Eintrag in Auswahlliste 981
 Farbe bereits geklickter Links 430
 Farbe Bits pro Pixel 1182, 1583, 1594
 Farbe Border 648, 659, 1014, 1057, 1063, 1069, 1581
 Farbe Button im Dialogfenster 981
 Farbe Button-Text im Dialogfenster 981
 Farbe deaktivierter Text im Dialogfenster 981
 Farbe der Scrolleiste 981
 Farbe des 3D-Rahmens 1014, 1058, 1063, 1069, 1581
 Farbe des Desktop 981
 Farbe des Dialogfensters 981
 Farbe des Dokumentfensters 981
 Farbe des Objektes 618, 1594
 Farbe eine VLINK Body 590, 1731
 Farbe einer Auswahlliste 981



Farbe eines 3D-Elements 981
 Farbe eines ALinks 1564
 Farbe eines ALinks Body 589
 Farbe eines Links Body 590, 1651
 Farbe eines Menü 981
 Farbe gerade angeklicktes 3D-Element 981
 Farbe Hintergrund 809, 818, 1918
 Farbe Hintergrund aktives Fenster 981
 Farbe Hintergrund Desktop 981
 Farbe Hintergrund Dokumentfenster 981
 Farbe Hintergrund Hintergrundbild das sich über die
 Hintergrundfarbe legt 809, 818, 1919
 Farbe inaktives Fenster 981
 Farbe Menüeintrag 981
 Farbe Menüleiste 981
 Farbe nichtaktive Fenstertitelzeile 981
 Farbe Rahmen 648, 659, 1014, 1057, 1063, 1069, 1581
 Farbe Rahmen Dokumentfenster 981
 Farbe selektierter Text in Auswahlliste 981
 Farbe Text 810, 820, 1928
 Farbe Text im Dokumentfenster 981
 Farbe Überschrift im Dialogfenster 981
 Farbe Überschrift in der aktiven Fenstertitelzeile 981
 Farbe Überschrift nichtaktive Fenstertitelzeile 981
 Farbe von noch nicht benutzten Links 427, 1651
 Farbe von Tooltip und Popuphilfe (Hint) 981
 Farbe Vordergrund 810, 820, 1928
 Farbenanzahl 1182
 Farbpalette 1182
 Farbtiefe 1182, 1583, 1594
 Favorit hinzufügen 1162, 1738
 Favoriten 862
 Favoriten-Datei Aufbau 862
 Favoriteneintrag 277, 278, 476, 477
 Favoritenliste 862, 1129, 1162, 1738, 1996
 Favoritenliste exportieren nach Netscape-Format 1164,
 1827
 Favoritenliste importieren aus Netscape-Format 1164,
 1827
 Favoritenverwaltung 1162, 1165, 1738, 1901
 Fehler bei Ausführung 1096
 Fehler bei Bildladen 1096
 Fehlerbedingung in JScript 98, 129
 Fehlerbehandlung 1147
 Fehlerbehandlung bei Javascript 114
 Fehlerbehandlung per onError 1147
 Fehlerbehandlung in JScript 97, 128
 Fehlermeldung beim Internet Explorer 66, 114
 Fehlermeldung wenn Webseite nicht gefunden 1163,
 1742
 Fehlermeldungen 129, 1100, 1147
 Fehlermeldungen zum Script 114
 Feld 864
 Feld (Collection) aller im Dokument befindlichen
 Objekte 256, 435, 629, 827, 1786
 Feld aller Anker im Dokument 448, 565
 Feld aller Applet-Objekte im Dokument 450, 570
 Feld aller Bookmarks 1108
 Feld aller eingebetteten Objekte im Dokument 451, 613,
 756
 Feld aller Elemente des Dokumentes 447
 Feld aller Elemente mit gemeinsamer URN 271, 274,
 450, 451, 452, 453, 455, 456, 471, 474, 566, 571, 614,
 632, 641, 642, 685, 707, 738, 742, 757, 763, 770,
 1000, 1040, 1046, 1190, 1913
 Feld aller Faforiten 1108

Feld aller Filter 511
 Feld aller Formular-Objekte 452, 641
 Feld aller Frames 453, 651
 Feld aller Frames im Fenster 382, 1627
 Feld aller HTML-Elemente mit gemeinsamen Tag-
 Namen 455, 685, 707, 1040
 Feld aller im Browser verfügbaren Mimetypen 1167
 Feld aller img Objekte 326, 454, 684, 707
 Feld aller Objekte mit HREF-Eigenschaft 455, 738
 Feld aller Objekte mit Link (HREF und AREA) 455, 738
 Feld aller styleSheet Objekte 999
 Feld aller Uniform Resource Name 504
 Feld aller URN 504
 Feld Anzahl der Elemente 166, 1650
 Feld der Cookies 177, 425, 489, 1598, 1781, 1784, 1789,
 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809,
 1811, 1814, 1816, 1818, 1820, 1822
 Feld der Objekt-Attribute-Referenzen 464
 Feld der Textknoten 268, 467
 Feld der Urls 1104
 Feld der Zeiger aller Layer 310
 Feld der Zeiger auf TextRectangle-Objekte 460, 483,
 564, 569, 574, 591, 600, 602, 611, 616, 620, 629, 683,
 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737,
 741, 746, 755, 762, 768, 777, 1021, 1030, 1034, 1038,
 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193,
 1780
 Feld dynamisch 71
 Feld Elemente anhängen 166, 1752, 1859
 Feld erstes Feldelement liefern und löschen 1899
 Feld Feldelemente physisch sortieren 168, 1906
 Feld für Eingabe aktivieren 634
 Feld für Eingabe deaktivieren 634
 Feld komplett nach String konvertieren 167, 1834
 Feld letztes Feldelement liefern und löschen 1855
 Feld Reihenfolge der Feldelemente physisch umkehren
 167, 1874
 Feld Werte dem Feld voransetzen 1912
 Feldelement Zeiger 452, 453, 454, 455, 456, 457, 461,
 630, 641, 642, 652, 685, 707, 738, 742, 770, 771, 998,
 1000, 1040, 1046, 1085, 1171, 1190, 1842
 Feldelemente Anzahl 451, 452, 453, 454, 455, 456, 457,
 461, 507, 511, 593, 641, 642, 652, 685, 707, 738, 742,
 759, 769, 771, 890, 940, 998, 999, 1000, 1040, 1046,
 1085, 1108, 1165, 1171, 1190, 1650
 Felder 864
 Feldlänge 451, 452, 453, 454, 455, 456, 457, 461, 507,
 511, 593, 641, 642, 652, 685, 707, 738, 742, 759, 769,
 771, 890, 940, 998, 999, 1000, 1040, 1046, 1085,
 1108, 1165, 1171, 1190, 1650
 Feldwert-Typen 1376
 Fenster 563, 573, 628, 736
 Fenster aktivieren Eventdurchreichung 401, 1886
 Fenster aktuell Referenz 649, 1687
 Fenster aktuell setzen 395, 1849
 Fenster aktueller Text der Statuszeile 384, 1699
 Fenster aktuelles 282
 Fenster als aktiv setzen 401, 1886
 Fenster Anzahl aller Frames bzw. IFrames 382, 1651
 Fenster anzeigen 395, 437, 1849
 Fenster Anzeigezustand 378, 1594
 Fenster auch schliessen, wenn opener geschlossen wird
 438
 Fenster Breite 1106
 Fenster des Browser rausschieben 294, 330, 395, 1839
 Fenster des Dokumentes schliessen 391, 1750



Fenster Dialogbox erzeugen 388, 392, 398, 1738, 1753, 1859
 Fenster Dialogfenster 383, 394, 395, 399, 416, 1681, 1839, 1842, 1872
 Fenster Dialogfenster Breite 381, 1610
 Fenster Dialogfenster Focus 408, 1903
 Fenster Dialogfenster Höhe 379, 1609
 Fenster Dialogfenster modal 405, 1901
 Fenster Dialogfenster nicht-modal 408, 1903
 Fenster Dialogfenster Position 380, 1609, 1610
 Fenster Dokument scrollen 400, 401, 1876
 Fenster Dokument verschieben 400, 401, 1876
 Fenster Eltern laut Fensterhierarchie 221, 383, 1667
 Fenster Elternfenster als Erzeuger 221, 382, 1666
 Fenster erzeugen 282, 437, 1849
 Fenster Eventdurchreichung aktivieren 401, 1886
 Fenster Fensterhierarchie 221, 383, 388, 1667, 1721
 Fenster Fenstertitel 395
 Fenster Focus 408, 1903
 Fenster Focus setzen 394, 1775
 Fenster Frame 381, 1627
 Fenster FRAME 665
 Fenster für Dialoge 1172
 Fenster für Meldungen 1172
 Fenster für Tooltip 1172
 Fenster ganz aus dem Bildschirm verschieben 394, 1839
 Fenster Größenveränderung 397, 439
 Fenster Höhe 1105
 Fenster IFrame 381, 1627
 Fenster IFRAME 665
 Fenster im Channel-Mode anzeigen 397, 438
 Fenster kann kein anderes überlagern 440
 Fenster laut Fensterhierarchie 221, 383, 388, 1667, 1721
 Fenster linker Rand 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Fenster Meldungsfenster erzeugen 388, 392, 398, 1738, 1753, 1859
 Fenster mit Eingabemöglichkeit 398, 1859
 Fenster modal 378, 1608
 Fenster modales Dialogfenster 405, 1901
 Fenster nicht in Windows-Taskleiste anzeigen 438
 Fenster nicht modal 378, 1608
 Fenster nicht-modales Dialogfenster 408, 1903
 Fenster oberer Rand 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Fenster oberstes 396, 438
 Fenster oberstes Fenster laut Fensterhierarchie 388, 1721
 Fenster oberstes in der Fenster-Hierarchie 282
 Fenster Objekte anzeigen 382, 1662
 Fenster Objekte im Offscreen rendern 382, 1662
 Fenster Objekte rendern 382, 1662
 Fenster öffnen 282, 395, 437, 1849
 Fenster physischer Fenstername 382, 395, 1659
 Fenster Popupfenster 392, 1178, 1757
 Fenster Popup-Fenster 1172
 Fenster Popup-Fenster anzeigen 1182, 1899
 Fenster Popup-Fenster Anzeigezustand 1181, 1643
 Fenster Popup-Fenster HTML-Dokument 1181, 1614
 Fenster Popup-Fenster schliessen 1181, 1825
 Fenster Position 383, 394, 395, 1684, 1839, 1842

Fenster schliessen 391, 397, 439, 1750
 Fenster schliesst sich selbst nach Wartezeit 292, 392, 1750
 Fenster Standard-Text der Statuszeile 378, 1607
 Fenster Statuszeile 378, 384, 1607, 1699
 Fenster Statuszeile Hyperlink-Text anzeigen 289, 386, 1702
 Fenster Statuszeile Scroller 287, 385, 1700
 Fenster Stautszeile Text Buchstabe für Buchstabe anzeigen 286, 384, 1699
 Fenster Titel 382, 1659
 Fenster Überlagerung 440
 Fenster und Kontextmenü 1162, 1656
 Fenster und sein Erzeuger 283
 Fenster und seine Eltern 283
 Fenster Verlauf 281, 283, 370, 382, 1633
 Fenster verschieben 1103
 Fenster Zeiger auf oberstes Fenster laut Fensterhierarchie 388, 1721
 Fenster Zeiger auf übergeordnetes Fenster laut Fensterhierarchie 221, 383, 1667
 Fenster Zustand der Anzeige 378, 1594
 Fenster: aktuelles 384, 1687
 Fenster: Druckbutton 398, 1859
 Fenster: Druckfenster aufrufen 398, 1859
 Fenster: Ereignisüberwachung 293
 Fenster: Fremdseite überwacht 293
 Fenster: Menüleiste anzeigen 397, 439
 Fenster: übergeordnetes 283
 Fensteraufblasen bei Auflösung von 640x480 299, 399, 1872
 Fensterauflösung ändern 299, 399, 1872
 Fensterbreite 397, 440, 1182
 Fensterelemente 392, 1178, 1757
 Fenstergröße ändern 1103
 Fenstergröße Veränderbarkeit 303
 Fenstergröße verändern 399, 1872
 Fenster-Handle 318, 324, 647, 658
 Fensterhierarchie 221, 383, 388, 395, 437, 1667, 1721, 1849
 Fenster-Hierarchie 282
 Fensterhöhe 397, 439, 1182
 Fensterinhalt 397, 440
 Fenstername 382, 1659
 Fenster-Objekt eines Objektes 648, 666, 1597
 Fensterrahmen 1103
 Fenstertitel 382, 395, 1659
 Fenstertitel des Dokumentes 428, 1714
 Fenstertitel permanant wechseln 428, 1714
 Fenstertiteltitel 382, 1659
 Fenstertitelzeile 981
 Fenstertitelzeile mit scrollendem Text 428, 1714
 feste, nicht mitscrollende Objekte 676
 Festes, nicht mitscrollendes Bild 676
 Festkomma-Darstellung 205, 1909
 Festplatte 864
 Festplattenordner der Webseite 24
 Festplattenverzeichnis unter Windows auslesen 1559, 1656
 fett 987, 989
 fieldSet Objekt 614
 fieldset Objekt Events 1134, 1955
 fieldset Objekt Filter 549
 fieldset Objekt Styles 832
 file 634, 685, 1722
 File 1108



File Upload-Control 695
 file:///c:/index.html 395, 1844
 File-Control 1722
 FileSystemObject Objekt 1430
 FileSystemObject.Drive Objekt 1434
 FileSystemObject.Drives Collection 1435
 FileSystemObject.File Objekt 1433, 1437, 1788
 FileSystemObject.Folder Objekt 1433, 1439, 1788, 1802
 FileSystemObject.Folder.Files Collection 1440, 1441, 1625
 FileSystemObject.Folder.Folders Collection 1441, 1442, 1703
 FileSystemObject.TextStream Objekt 1442
 FILEUPLOAD 1095, 1096
 filter 820
 Filter 548, 749, 870, 878
 Filter a Objekt 548
 Filter Alpha 518
 Filter AlphaImageLoader 520
 Filter Animation starten 518, 2008
 Filter Animation stoppen 518, 2008
 Filter Anwendbarkeit 512, 2001
 Filter Anzeige zurücksetzen 518, 2008
 Filter applet Objekt 548
 Filter area Objekt 548
 Filter Barn 520
 Filter BasicImage 514, 521, 2003
 Filter BlendTrans 522
 Filter Blinds 523
 Filter Blur 514, 523, 2004
 Filter body Objekt 548
 Filter button Objekt 549
 Filter CheckerBoard 524
 Filter Chroma 525
 Filter Compositor 525
 Filter currentStyle Objekt 549
 Filter custom Objekt 549
 Filter Dauer der Animation 512, 2001
 Filter Deckungskraft 512, 514, 517, 2002, 2003, 2006
 Filter des Internet Explorer 810, 820, 1931
 Filter div Objekt 549
 Filter document Objekt 549
 Filter dokumentbezogen 509
 Filter DropShadow 526
 Filter dynamisch 508
 Filter eines HTML-Elementes 508
 Filter Emboss 526
 Filter Engrave 526
 Filter Fade 527
 Filter Feld aller Filter 511
 Filter fieldset Objekt 549
 Filter FlipH 531
 Filter FlipV 531
 Filter font Objekt 550
 Filter form Objekt 550
 Filter frame Objekt 550
 Filter frameset Objekt 550
 Filter Glow 531
 Filter Gradient 531
 Filter GradientWipe 513, 532, 2003
 Filter Grauskala 512, 2002
 Filter Gray 533
 Filter ICMFilter 533
 Filter iframe Objekt 550
 Filter im Internet Explorer 508
 Filter img Objekt 551

Filter Initialisierung 518, 2008
 Filter input button Objekt 551
 Filter input checkbox Objekt 551
 Filter input file Objekt 551
 Filter input hidden Objekt 552
 Filter input image Objekt 552
 Filter input Objekt 551
 Filter input password Objekt 552
 Filter input radio Objekt 552
 Filter input reset Objekt 552
 Filter input submit Objekt 553
 Filter input text Objekt 553
 Filter Inset 533
 Filter Invert 534
 Filter Iris 534
 Filter Komplementärfarben 512, 2002
 Filter Light 535
 Filter lineare Transformation 512, 2001
 Filter marquee Objekt 553
 Filter MaskFilter 535
 Filter Matrix 515, 536, 2005
 Filter Matrixfeld 513, 2002
 Filter MotionBlur 537
 Filter Nummer der Komposition 512, 2002
 Filter object Objekt 553
 filter Objekt 508, 810, 820, 878, 1931
 Filter option Objekt 553
 Filter Pixelate 537
 Filter RadialWipe 517, 538, 2007
 Filter RandomBars 539
 Filter RandomDissolve 540
 Filter RevealTrans 540
 Filter RevealTrans bei IE 4.x und ab IE 5.5 510
 Filter runtimeStyle Objekt 553
 Filter select Objekt 554
 Filter selection Objekt 554
 Filter Shadow 541
 Filter Slide 542
 Filter span Objekt 554
 Filter Spiral 543
 Filter Start 518, 2008
 Filter Status der Filteranimation 516, 2006
 Filter stoppen 518, 2008
 Filter Stretch 544
 Filter Strips 514, 545, 2003
 Filter style Objekt 554
 Filter table Objekt 554
 Filter table.caption Objekt 554
 Filter table.col Objekt 554
 Filter table.colGroup Objekt 555
 Filter table.tBody Objekt 555
 Filter table.tFoot Objekt 555
 Filter table.tHead Objekt 555
 Filter table.tr Objekt 555
 Filter table.tr.td Objekt 555
 Filter table.tr.th Objekt 555
 Filter textarea Objekt 555
 Filter var Objekt 555
 Filter Wave 546
 Filter Wheel 546
 Filter window Objekt 556
 Filter Xray 547
 Filter Zeiger 509
 Filter ZigZag 547
 filter: progid:DXImageTransform.Microsoft 509
 filter:shadow() 542



Filter-Animation eines Objektes 508
 Filter-Attribut des Style-Objekt 510
 Filterbezeichner vom IE 4.x und ab IE 5.5 509
 Filter-Effekte 1100
 Filtereigenschaften 511
 Filtermethoden 518
 Filternummer 517, 2007
 Filterreferenz 509
 filters Collection 511
 filters[0].Apply(); 528, 672
 filters[0].Play(); 528, 672
 Firewall und Cookies 486
 fivePoint 1703, 1723
 fixed 984
 Fließkomma 57
 FlipH Filter 531
 FlipV Filter 531
 float 818
 Float 1376
 Floating point 57
 Flussrichtung bei Objektanzeige 816, 826, 1949
 Focus 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 683, 688, 693, 698, 705, 710, 715, 719, 723, 727, 732, 740, 746, 754, 761, 767, 776, 1019, 1030, 1043, 1049, 1054, 1060, 1066, 1072, 1077, 1078, 1079, 1119, 1193, 1635, 1679, 1748, 1986
 Focus eines Objektes 424, 1560
 Focus im Dokument 424, 1560
 Focus input Objekt 690, 694, 699, 707, 711, 716, 720, 725, 729, 1883
 Focus setzen 435, 483, 563, 569, 574, 591, 599, 611, 616, 620, 629, 650, 661, 668, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 746, 755, 761, 777, 1021, 1030, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1193, 1775
 Focus wegnehmen 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 649, 660, 668, 683, 688, 693, 697, 705, 710, 714, 719, 723, 727, 732, 745, 754, 761, 776, 1019, 1030, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1193, 1745
 focus() 638
 Focus-Event 435, 483, 563, 569, 574, 591, 599, 611, 616, 620, 629, 650, 661, 668, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 746, 755, 761, 777, 1021, 1030, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1193, 1775
 Focussierbarkeit Body 589, 614, 759
 Focussierbarkeit Objekt 481, 561, 567, 572, 609, 618, 627, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 744, 753, 775, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1191, 1632
 Focus-Zustand 611
 fokussieren 1097
 fokussieren HTML-Element 1100
 Folge aller Layer 329
 Folien 326
 Font 618, 980, 987
 FONT 1245
 FONT COLOR 216, 1775
 Font Familie 618, 1625
 Font Größe 619, 1691
 font Objekt 618
 font Objekt Events 1134, 1955
 font Objekt Filter 550
 font Objekt Größe 619, 1691
 font Objekt Styles 833

font Objekt Typ 618, 1625
 FONT SIZE 216, 1775
 Font Typ 618, 1625
 Font-Dateien 982
 Fontglättung auf Bildschirm 1183, 1627
 Fontgröße 619, 1691
 FontName 277, 476
 FontSize 277, 476
 FOR 41, 1095
 for .. in 174
 for Anweisung 87
 for in Anweisung 88, 150, 209, 1676
 ForeColor 277, 476
 Form des Bildausschnittes 993
 Form eines Objektes 562, 573, 1688
 Form feed 56, 58, 114
 Form Feed 56, 58, 113, 114
 form Objekt 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 1607
 form Objekt Events 1134, 1956
 form Objekt Filter 550
 form Objekt Styles 833
 form.input image Objekt 271, 471
 FormatBlock 277, 476
 formatierende Tags 278, 477
 Formdata 562, 572, 1167, 1685
 Formular 325, 448, 450, 451, 452, 511, 566, 571, 613, 626, 627, 641, 670, 686, 691, 695, 696, 700, 703, 708, 712, 713, 717, 721, 725, 730, 757, 767, 1103, 1104, 1149, 1558, 1607, 1711, 1721
 Formular abschicken 638, 1104
 Formular Autovervollständigung 627, 708, 709, 725, 727, 1572, 1731
 Formular Button 593
 Formular Button Submit 721
 Formular Daten permanent speichern 1163, 1742
 Formular Daten speichern 1163, 1742
 Formular Eingabe 725
 Formular Laufschrift 640
 Formular löschen 637
 Formular mit Autocomplete 1162
 Formular mit Autovervollständigung 1162, 1163, 1742
 Formular Optionswert zum Senden 767, 1711
 Formular Password 707
 Formular reset 630, 1127, 1870, 1994
 Formular Reset-Button 716
 Formular rücksetzen 626, 1103
 Formular senden 626, 627, 638, 702, 1131, 1621, 1999
 Formular senden Optionswert 767, 1711
 Formular senden Wortumbruch 1078, 1733
 Formular sofort abschicken 638
 Formular submit 631, 1131, 1908, 1999
 Formular Submit-Button 721
 Formular Usereingabe 725
 Formular versteckte Daten 622
 Formular Wortumbruch in Textarea 1078, 1733
 Formular Wortumbruch senden 1078, 1733
 Formular Zeiger auf Element 597, 614, 633, 634, 635, 636, 637, 638, 641, 686, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 753, 759, 765, 1077, 1627
 Formular zurücksetzen 1127, 1994
 Formulardaten per Cookies 485
 Formulare Auslesen eines Eingabefeldes 639
 Formulare Wert einem Eingabefeld zuweisen 640
 Formulkomponenten 452, 641
 formular-orientierte Datenbeschaffung 622



Formularprüfung 626
 Fortschrittangabe entlang der Timeline 1676
 Forward Button 863
 forward() 1165
 Forward-Button 1775
 fr 1659
 Fragezeichen 988
 frame 752, 1579
 Frame 283, 293, 361, 381, 396, 438, 453, 563, 573, 628, 651, 736, 1145, 1627
 FRAME 312, 318, 642, 652, 1096, 1097, 1248
 Frame aktuell Referenz 649, 1687
 Frame Breite 659, 1106, 1595
 Frame Größenänderung 649, 1662
 Frame Höhe 660, 1105, 1682
 Frame inline 453, 651
 frame Objekt 453, 642, 651, 665
 frame Objekt Events 1135, 1956
 frame Objekt Filter 550
 frame Objekt Styles 833
 FRAME Reload eines Dokumentes mit allen seinen
 FRAMES 315, 321, 644, 655
 Frame und Datenaustausch 316, 322, 646, 656
 Frame und Status im Rahmen des Browser-Sicherheitsmodells 648, 666, 1566
 Frame Zeiger 381, 1627
 Framedarstellung und Abmahnungsgefahr 319, 643, 653
 Framedarstellung und rechtliche Zulässigkeit 319, 643, 653
 Frameinhalte 315, 321, 645, 655
 Frame-Objekt 1095, 1096
 Frames Anzahl 382, 1651
 frames Collection 665
 Frames Zwischenraum 659, 1627
 frames[] 315, 321, 645, 655
 FRAMESET 312, 318, 642, 648, 652, 750, 1095, 1096, 1097, 1187, 1250
 FRAMESET nicht vom Browser erkannt 750
 frameset Objekt 652
 frameset Objekt Events 1135, 1956
 frameset Objekt Filter 550
 frameset Objekt Styles 834
 Framset 1101
 Französisch 1659
 freeze 887, 895, 900, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1642, 1984
 frei definierbare Daten nicht permanent speichern 863, 864
 frei definierbare Daten permanent speichern 862
 frei definierbarer Datentyp 62
 frei deklarierte Funktion 106
 fremde Seite 283, 361, 1145
 fremde Webseite 313, 320, 643, 653
 fremder Frame 293, 1146
 Fremderweiterungen der Browserfähigkeiten 1145
 Fremdes Dokument laden ohne Framedarstellung 313, 319, 643, 653
 fremdes HTML-Dokument 361, 1145
 Fremdseite 293
 Fremdseite kann Seite überwachen 293
 fromCharCode(..) 1100, 1152, 1155
 fromLeft 1704, 1723
 Fromular Daten permanent speichern 1162
 FTP 562, 572, 1108, 1167, 1675
 FTP-Zugriffschutz 51
 fullscreen 397, 439

function 54, 77, 81, 143
 function Anweisung 88, 95, 102, 105, 106, 161, 1553, 1567, 1589, 1650, 1739, 1745
 function Basis-Datenstruktur 56
 function Datentyp 56
 Function JScript-Objekt 149
 Function Objekt 95, 149, 160
 Function Script-Objekt 56, 88, 89, 102, 105, 106, 160, 161, 194, 1553, 1650
 Funktion 102, 160
 Funktion als Methode 102
 Funktion als Objektkonstruktor 111
 Funktion Anweisung return 95, 105
 Funktion Anzahl der Argumente 162, 195, 1650
 Funktion Anzahl der Parameter 195, 1650
 Funktion Anzahl Parameter 104, 160
 Funktion Argument und Argumentenliste 103
 Funktion Argumente und Parameter 103
 Funktion Argumentenliste 88, 106
 Funktion Aufruf 88, 103, 106
 Funktion aufrufen 195, 1589
 Funktion einlesen der Werte der Argumente aus dem Funktionsaufruf 104, 161, 162, 1553, 1650
 Funktion frei deklariert 103, 106
 Funktion Gültigkeit 88, 106
 Funktion Gültigkeit von Variable 95, 105
 Funktion Kopf 88, 106
 Funktion optionale Argumente und Parameter 103
 Funktion Parameter 103
 Funktion Parameter und Parameterliste 104
 Funktion Parameterliste 88, 106
 Funktion privat 106
 Funktion Rekursion 161, 1589
 Funktion return Anweisung 95, 105
 Funktion Rumpf 88, 106
 Funktion Script-Objekt 1567, 1589, 1650, 1739, 1745
 Funktion und Abarbeitungsfolge 112
 Funktion und Funktionswert 103, 104
 Funktion und Methode 102
 Funktion und optionaler Funktionswert 104
 Funktion und Prototyping 103
 Funktion und Rekursion 112
 Funktion und Rekursionen 113
 Funktion und Variablenübergabe 113
 Funktion Verschachtelung 88, 106
 Funktion vordefiniert 106
 Funktion vordefiniert oder frei deklariert 103
 Funktion Wert des Argumentes 161, 1553
 Funktion Wert liefern 95, 105
 Funktion Werte der Argumente aus dem Funktionsaufruf einlesen 104, 161, 162, 1553, 1650
 Funktion Zeiger auf Aufrufer der Funktion 195, 1589
 Funktionsaufruf mit Parameterliste 104
 Funktionskopf 103, 106
 Funktionsrumpf 104, 106, 161, 1589
 Funktionsrumpf Zeiger 161, 1589
 Funktionswert 95, 105
 Funktionswert der Funktion 104
 Funktionswert einer Funktion 103
 Fußnote 983, 998, 1553, 1678
 Fusszeile des Dokumentes 419
 fzeichenfolge 1184
 g 1185
 GALLERYIMG 749
 Ganzzahldivision 75
 ganzzahlige Division 198, 203



ganze Zahl 199, 1745, 1775
 Ganzzahl 57
 Ganzzahldivision 75
 Ganzzahl-Division 78
 ganzzahlig konvertieren 197, 203
 ganzzahlig teilbar konvertieren 198, 203
 gekapselte Daten 223
 gemeinsam übergeordneter Layer 329
 gemeinsamen Tag-Namen 455, 685, 707, 1040
 Genre 874, 1469
 geniertes ID des Objektes 239
 gesamter Plain-Text im Objekt 610, 619, 775
 geschütztes Blank 816, 826, 1948
 Geschwindigkeit Widergabe 885, 890, 893, 898, 905,
 909, 915, 922, 927, 932, 946, 954, 966, 971, 1693
 Gestalt eines Objektes 562, 573, 1688
 get 622, 1656
 GET 1559, 1656
 getimtes Element 1713
 getTime() 420, 1987, 2000
 gi 1185
 GIF 671, 702, 983, 988
 Gif-Bild 887, 911, 917, 919, 924, 930, 934, 948, 957,
 973, 1120, 1121, 1987, 1988
 Gif-Datei 907, 1101
 GIF-Datei 508
 Gleichheitszeichen 984
 Gleitkommazahl 57
 globale Gültigkeit 65
 Glow Filter 531
 GMT 873
 go(position) 1165
 Gradient Filter 531
 GradientWipe Filter 513, 532, 2003
 Grafik 983
 Grafik vorladen 675
 Grafikdatei für Hintergrund 327
 Grafik-Smarttag ab IE 6.x 749
 Grafiksystem 894, 898, 905, 1728
 Graphics Interchange Format 671, 702
 Grauskala Filter 512, 2002
 Grauzustand (Dimmed) und Selektiertheit des Checkbox-
 Control 691, 1635
 Gray Filter 533
 Greenwich-Time 489
 groove 983, 987, 991
 groß 987, 989
 Gross oder Klein 1101
 Groß und Klein bei der Suche 1185
 Großbuchstabe 987, 990, 1100
 Grossbuchstaben 1101
 Größe des Dokumentes 427, 681, 1625
 Größe Objekt 1128, 1995
 Größenänderung des Fensters 1103
 Größenänderung Frame 649, 1662
 Größenveränderung des Fensters 397, 439
 größere der beiden Zahlen 199, 1838
 Größter gemeinsamer Teiler zweier ganzer Zahlen 197,
 202
 groups 1017, 1683
 Grün-Anteil 980
 Gültigkeit der Zeiger 59
 Gültigkeit einer Variable 65
 Gültigkeit Funktion 88, 106
 Gültigkeit global 65
 Gültigkeit lokal 65

Gültigkeit von Variable in Funktion 95, 105
 h:m:s.f 872
 halten in der Timeline 1642
 hand 993
 Handle 318, 324, 647, 658
 Handler überschreiben 291
 Handlerhierarchie 292, 1144, 1145
 hash 734
 Hauptspeicher 58
 Hauptverzeichnis 50
 Hauptverzeichnis des Webs auf dem Server 49
 HEAD 223, 736, 748, 752, 772, 1251
 head Objekt Events 1135, 1956
 head Objekt Styles 834
 HEAD-Abschnitt des Dokumentes 40
 height 397, 439, 671
 HEIGHT 306, 309, 325, 450, 451, 566, 570, 613, 670,
 671, 756
 help 406, 408, 988, 993
 Helpdatei 405, 1901
 Hersteller CPU 411, 854, 1601
 Hersteller Prozessor 411, 854, 1601
 hexa 57, 980
 Hexa 421
 hexadezimale Zahl 57
 HexaKette 421
 hidden 634, 685, 1722
 HIDDEN 309, 451, 581, 613, 756
 Hidden-Control 1722
 hide 329
 Hierarchie der HTML-Elemente 222
 Hierarchie der Objekte 220, 222
 Hierarchie im Dokument 222
 Hierarchie im HTML-Dokument 222
 HIGHER 941
 Hilfe F1 1119, 1986
 Hilfe im Browser 1100
 Hilfedatei 405, 1901
 Hintergrund 327, 990
 Hintergrund aktives Fenster 981
 Hintergrund ausblenden 420
 Hintergrund auswählen 422
 Hintergrund Dokumentfenster 981
 Hintergrund ein- bzw. ausblenden 420
 Hintergrund eines Objektes 809, 818, 1918
 Hintergrund transparent 328, 990
 Hintergrundbild 809, 818, 979, 1918
 Hintergrundbild beim IE als Wasserzeichen 1918
 Hintergrundbild das sich über die Hintergrundfarbe legt
 809, 818, 1919
 Hintergrundbild eines Objektes 1014, 1027, 1041, 1063,
 1069, 1574
 Hintergrundbild fixieren 990
 Hintergrundbild im Body 589
 Hintergrundbild im Dokument 1574
 Hintergrundbild kacheln 809, 819, 1919
 Hintergrundbild scrollen 1581
 Hintergrundbild scrollen Body 589
 Hintergrunddatei 983
 Hintergrundfarbe 277, 476, 809, 818, 983, 1918
 Hintergrundfarbe des Dokumentes 424, 589, 743, 1581
 Hintergrundfarbe des Layers 328
 Hintergrundfarbe Hintergrundbild das sich über die
 Hintergrundfarbe legt 809, 818, 1919
 Hintergrundgrafik 983
 Hintergrund-Grafik 983



Hintergrundbild Position 809, 819, 1919
 Hintergrundmusik 575
 Hintergrundmusik Balance 1574
 Hintergrundmusik Wiederholungen 582, 681, 703, 1653
 Hints 981
 hinzufügen DHTML-Eigenschaft 249, 482, 563, 568, 573, 591, 599, 602, 610, 615, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 hinzufügen Eigenschaft 249, 482, 563, 568, 573, 591, 599, 602, 610, 615, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 hinzufügen zum Clipboard 1115, 1981
 History 437, 736, 863, 1166, 1167, 1739, 1869
 history Collection 1165
 history Objekt 281, 283, 370, 382, 1165, 1167, 1633, 1739, 1869
 history.go(0) 1166, 1824, 1995
 History-Eintrag des Dokumentes ersetzen 1167, 1869
 History-Eintrag ersetzen 397, 440
 HKEY_LOCAL_MACHINE\software\microsoft\internet explorer\main\urltemplate 1163, 1742
 HMTL-Text im Textbereich 459, 1083, 1634
 Höhe des Arbeitsbereiches 854, 1574
 Höhe des Arbeitsbereiches auf dem Bildschirm ohne Windows-Taskbar 1182, 1574
 Höhe des Objektes 648, 666, 681, 744, 753, 1016, 1058, 1064, 1070, 1632
 Höhe des vertikalen Scrollbereiches 482, 562, 568, 598, 610, 628, 663, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 744, 750, 754, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1684
 Höhe des vertikalen Scrollbereiches Body 590, 615, 760, 766
 Höhe Frame 660, 1682
 Höhe in Pixel vom Fenster 1105
 Höhe in Pixel vom Frame 1105
 Höhe minimale Höhe des Objektes 812, 822, 1937
 Höhe Objekt 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 811, 813, 817, 820, 823, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592, 1932, 1940
 hold 887, 895, 900, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1642, 1984
 holding 866
 Homepage 856
 Homepage Domain 856, 857, 1830
 Homepage laden 857, 1845
 horizontal 1703, 1723
 horizontale Bildpunkte 1183, 1607, 1652
 horizontale Linie 277, 325, 476, 670
 horizontaler Scrollbereich 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 horizontaler Tabulator 56, 58, 113, 114
 host 734
 Host der Webseite 24

hostname 25, 734
 Hostname und Port 561, 572, 1167, 1633
 hostname:port 561, 572, 1166, 1167, 1633
 hotkeys 439
 HP Unix 412, 1672
 HP-UX 412, 1672
 href 734
 HREF 289, 306, 332, 386, 448, 455, 565, 571, 734, 738, 739, 1096, 1598, 1702
 HREF="javascript:void(0); 77
 HREF-Attribut 455, 738
 HREF-Wirkung 572, 1662
 hspace 1016, 1627
 hspace 671
 HSPACE 306, 325, 450, 566, 570, 670, 671
 htaccess und httpasswd und Passwordschutz 51
 htaccess und Passwordschutz 51
 HTC 850
 HTC-Datei 337, 490, 499, 501, 502, 503, 504, 781, 865, 1755, 1770, 1772, 1981, 1983, 1993
 HTC-Datei importieren 492
 HTC-Datei und HTML-Dokument 492
 HTC-Datei Aufbau 490
 HTC-Datei Container 492
 HTC-Datei Events 503
 HTC-Datei HTC-Komponente Container 492
 HTC-Datei HTC-Komponente Eigenschaft 492
 HTC-Datei HTC-Komponente Event 492
 HTC-Datei HTC-Komponente Eventhandler 492
 HTC-Datei HTC-Komponente Methode 492
 HTC-Datei Methoden 499
 HTC-Datei PUBLIC:ATTACH 492
 HTC-Datei PUBLIC:COMPONENT 493
 HTC-Datei PUBLIC:DEFAULTS 494
 HTC-Datei PUBLIC:EVENT 495
 HTC-Datei PUBLIC:METHOD 497
 HTC-Datei PUBLIC:PROPERTY 497
 HTC-Komponente 492
 HTC-Komponente Aufbau 492
 HTC-Komponente Eigenschaft 492
 HTC-Komponente Event 492
 HTC-Komponente Eventhandler 492
 HTC-Komponente Methode 492
 HTML 1108, 1253
 HTML 4.01 976
 HTML 4.x 224
 html comment Objekt 665
 html comment Objekt Events 1135, 1956
 html comment Objekt Styles 835
 HTML Component (HTC) Datei 850
 HTML Element animieren 907
 html Objekt 418
 html Objekt Events 1135, 1956
 html Objekt Styles 835
 HTML-Attribut entfernen 238
 HTML-Attribut Funktionsaufruf anstelle des Wertes 103
 HTML-Attribut Wert 254, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 750, 755, 761, 768, 777, 827, 849, 850, 863, 864, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1777
 HTML-Attribut Wert aus Funktionsaufruf 103
 HTML-Attribute 259, 484, 564, 569, 575, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741,



747, 751, 755, 762, 768, 778, 827, 850, 863, 864,
1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067,
1073, 1081, 1190, 1194, 1866

HTML-Attribute eines Objektes 250, 483, 563, 568,
573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663,
668, 683, 688, 693, 697, 701, 705, 710, 715, 719, 723,
727, 732, 737, 740, 746, 754, 761, 767, 776, 826,
1019, 1030, 1034, 1038, 1043, 1048, 1054, 1060, 1066,
1072, 1079, 1189, 1193, 1746

HTML-Code im Textbereich 460, 827, 1084, 1853

HTML-Code und/oder Script-Code einfügen 240, 258,
484, 564, 574, 592, 600, 603, 612, 617, 621, 630, 661,
669, 684, 689, 694, 698, 701, 706, 711, 715, 720, 724,
728, 733, 741, 746, 762, 768, 777, 827, 1031, 1067,
1073, 1081, 1194, 1829

HTML-Container für ein Bild 671

HTML-Datei 864

HTML-Dateien auf dem Server 48

HTML-Dokument 438, 864, 1165

HTML-Dokument als Datei speichern 1129, 1996

HTML-Dokument Autoscan einer Url 1163, 1742

HTML-Dokument Baumhierarchie 222

HTML-Dokument Daten-Insel 1195

HTML-Dokument Datenmenge 1195

HTML-Dokument drucken 398, 1859

HTML-Dokument Elemente-Hierarchie 222

HTML-Dokument Fenster schliessen 391, 1750

HTML-Dokument im DOM 418

HTML-Dokument im Layer 329

HTML-Dokument im Popup-Fenster 1181, 1614

HTML-Dokument Lade-Ereignisse 1162

HTML-Dokument laden 395, 1844

HTML-Dokument Layout auf einen Schlag ändern 994

HTML-Dokument linke obere Ecke 1104

HTML-Dokument mit all seinen Elementen laden 1101

HTML-Dokument mit embedded Objekte laden 369,
1168

HTML-Dokument mit Plugins laden 369, 1168

HTML-Dokument mit Sprachausgabe 1498

HTML-Dokument mit Style-Sheet 976

HTML-Dokument neu laden 1103

HTML-Dokument nicht gefunden 1163, 1742

HTML-Dokument Objekthierarchie 222

HTML-Dokument Rumpf 584

HTML-Dokument Styles auf einen Schlag ändern 994

HTML-Dokument Textsuche 1164, 1844

HTML-Dokument und Browserfenster 220

HTML-Dokument und Ebenen 326

HTML-Dokument und HTC-Datei 492

HTML-Dokument Userdaten permanent sichern 976,
1624, 1837, 1874

HTML-Dokument Userdaten permanent speichern 976,
1624, 1837, 1874

HTML-Dokument Userdaten sichern 976, 1624, 1837,
1874

HTML-Dokument Userdaten speichern 976, 1624, 1837,
1874

HTML-Dokument verlassen 1104, 1129, 1996

HTML-Dokument Verlauf 281, 283, 370, 382, 1633

HTML-DOM 232, 239, 1825

HTML-DOM attributes Collection 266, 464

HTML-DOM beim Internet Explorer 238

HTML-DOM beim Netscape 6.x 232

HTML-DOM childNodes Collection 268, 467

HTML-DOM children Collection 271, 471

HTML-DOM Collection attributes 266, 464

HTML-DOM Collection childNodes 268, 467

HTML-DOM Collection children 271, 471

HTML-DOM Collection tags 275, 474

HTML-DOM Collectionen 266

HTML-DOM Eigenschaften 242

HTML-DOM Methoden 249

HTML-DOM tags Collection 275, 474

HTML-DOM Übersicht 238

HTML-Element 268, 467, 1096

HTML-Element animieren 508

HTML-Element beim IE und NS 304, 418

HTML-Element beim NS und IE 304, 418

HTML-Element deaktiviere 1095

HTML-Element erzeugen 222, 587

HTML-Element Erzeugung durch Makro 275, 474

HTML-Element fokussieren 1100

HTML-Element im DOM 418

HTML-Element mit transparentem Hintergrund 990

HTML-Element scrollen 1103

HTML-Element und Filter 508

HTML-Element verändern 1103

HTML-Element verlassen 1095

HTML-Element-Dimension 984

HTML-Elemente Hierarchie 222

HTML-Elemente mit gemeinsamen Tag-Namen 455,
685, 707, 1040

HTML-Elemente-Anordnung im Dokument 985

HTML-Element-Lade-Eventbehandlung 1161

HTML-Element-Sichtbarkeit 985

HTML-Kommentar 665, 1255

HTML-Namensraum 504

HTML-Objekt Attribute 463

HTML-Programmierung dynamisch 225

HTML-Schnappschuss des Dokumentes 864

HTML-Seite ein- bzw. ausblenden 749

HTML-Seite mit Frame 361, 1145

HTML-Seite verlassen 1097, 1104

HTML-Seite wechseln 1104

HTML-Seiten-Refresh für WebCam 1635

HTML-Standard 976

HTML-Tag 418

HTML-Tag Event Kodierung 354, 1093

HTML-Tag Objekt 418

HTML-Tag Referenz 61

HTML-Tag Zeiger 61

HTML-Tag-Bezeichner 240

HTML-Tag-bezogene Eigenschaften 984

HTML-Tags 240

HTML-Tags deprecated 224

HTML-Tags im Objekt 242, 481, 560, 567, 571, 597,
601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691,
695, 700, 703, 708, 712, 717, 721, 725, 730, 736, 739,
743, 750, 751, 752, 759, 765, 774, 1014, 1028, 1032,
1036, 1041, 1046, 1051, 1058, 1063, 1069, 1076, 1187,
1191, 1195, 1590

htpasswd und Passwordschutz 51

HTTP 562, 572, 1108, 1167, 1675

HTTP Header-Transaktion 567, 753, 1594

HTTP Response Header 750, 1634

HTTP User-Agent 412, 1726

http-equiv 509

HTTP-EQUIV 748

HTTP-Methoden eines Objektes 561, 1656

HTTPS 1108

HTTP-Server 24

HTTPS-Protokoll 489



HTTP-Verzeichnis anzeigen 857
 HTTP-Verzeichnis im aktuellen Fenster anzeigen 857, 1844
 HTTP-Verzeichnis im Fenster nach Wahl anzeigen 858, 1845
 Hurenkind 987
 Hyperlink 278, 289, 386, 477, 1702
 Hypertext-Link 559
 i 1185
 i UND g 1185
 ICMFilter Filter 533
 ICO-Datei 420, 1634, 1679
 IconFile 862
 IconIndex 862
 ID 222, 224, 239, 255, 256, 267, 435, 436, 466, 467, 483, 507, 560, 561, 564, 567, 569, 574, 582, 592, 597, 600, 605, 609, 611, 614, 616, 620, 627, 648, 650, 659, 661, 664, 666, 668, 681, 683, 686, 687, 691, 696, 700, 703, 708, 713, 717, 721, 725, 726, 730, 732, 736, 737, 739, 741, 743, 746, 753, 759, 760, 762, 774, 777, 807, 827, 977, 978, 999, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1076, 1077, 1080, 1103, 1189, 1193, 1603, 1659, 1786, 1787, 1799, 1868
 ID Body 589, 614, 759, 765
 ID des Label 730, 1634
 ID des Objektes 239, 247, 430, 563, 568, 573, 590, 599, 602, 615, 619, 628, 649, 660, 663, 667, 682, 688, 692, 697, 701, 704, 709, 714, 718, 722, 727, 731, 737, 740, 745, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1724
 ID internes 481, 561, 567, 572, 597, 601, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 ID Objekt 481, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 ID.eigenschaft 393, 1178
 ID_Player Objekt 1478
 ID_Player.cdromCollection Collection 1483
 ID_Player.closedCaption Objekt 1484
 ID_Player.controls Objekt 1484
 ID_Player.currentMedia Objekt 1486
 ID_Player.currentPlaylist Objekt 1488
 ID_Player.error Objekt 1491
 ID_Player.mediaCollection Collection 1491
 ID_Player.network Objekt 1493
 ID_Player.playlistCollection Collection 1494
 ID_Player.settings Objekt 1496
 Identität in Wert und Datentyp 78
 IE 772
 IE 5.5 393, 1178, 1757
 IE 5.x Webseite mit eigenem Icon 419, 1634, 1679
 IE 6.0 unter Windows XP 412, 1726
 IE 6.x und Plugins 369, 1169, 1170
 IE 6.x und Smart-Tags 749
 IE 6.x-Inkompatibilitätsmodus zu seinen Vorgängern 802
 IE 6.x-Kompatibilitätsmodus zu seinen Vorgängern 802
 IE Abschaltung der automatischen Umrandung von angeklickten Objekten 1775
 IE ActiveX-Control anstelle Plugin 451, 613, 756, 1171

IE Active-X-Komponenten 854
 IE bedingtes Parsen 53, 99
 IE Behavior 850
 IE CSS und Attribute width und hight 803
 IE CSS1- Inkonformität 801
 IE CSS1- Kompatibilität auch bei Frameset-Dokument 802
 IE CSS1- Kompatibilität außer in einem Frameset-Dokument 802
 IE CSS1- Konformität genau im Umfang von HTML 4 laut World Wide Web Consortium 802
 IE CSS1-Konformität 801
 IE CSS-Konformität der Versionen 801
 IE Hintergrundbild als Wasserzeichen 1918
 IE Parsen bedingt 53, 99
 IE Pars-Reihenfolge 807
 IE Plugin -Ersatz durch ActiveX-Control 451, 613, 756, 1171
 IE Reload des Dokumentes nach Resize des Browserfenster 1166, 1824, 1995
 IE Scriptmaschine Buildnummer 40, 1875
 IE Scriptmaschine Hauptversion 40, 1875
 IE Scriptmaschine Sprache 40, 1875
 IE Scriptmaschine Unterversion 40, 1876
 IE Standard-Behavior 850
 IE Standard-Eigenschaften 249, 482, 563, 568, 573, 591, 599, 602, 610, 616, 619, 628, 649, 660, 663, 667, 682, 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731, 737, 740, 745, 750, 751, 754, 761, 767, 776, 1019, 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1196, 1737
 IE Style-Eigenschaften, die nur im HEAD des Dokumentes ansprechbar sind 818
 IE Style-Eigenschaften, die nur per Script ansprechbar sind 816
 IE Style-Eigenschaften, die nur per STYLE-Attribut ansprechbar sind 818
 IE Style-Eigenschaftenübersicht zu Script- und STYLE-Kodierung 818
 IE und Sound 575
 if ! IE 5 39
 if else Anweisung 91
 if IE 5 38, 39
 iframe 752, 1579
 IFrame 381, 1627
 IFRAME 277, 453, 476, 651, 1101, 1108
 iframe Objekt 665, 816, 826, 1950
 iframe Objekt Events 1135, 1956
 iframe Objekt Filter 550
 iframe Objekt Styles 835
 IFRAME ohne Fensterrahmen 665
 IFRAME Start von Programmen 1685
 IFRAME Transparenz 665
 IFRAME Überlappung 666
 IFrame Zeiger 381, 1627
 IFrames Anzahl 382, 1651
 ILAYER 326
 im Dokument ZUERST gefundene Objekt 255, 435, 827, 1786
 image 325, 670, 685, 1097, 1722
 Image 277, 476, 563, 737, 754, 1108, 1721
 Image Download 1112, 1977
 Image Hintergrund Position 809, 819, 1919
 Image in geringerer Auflösung Url 681, 703, 1654
 Image in normaler Auflösung 1195
 Image Lade-Ereignisse 1161



Image Tooltip 571, 680, 686, 703, 1564
 Image vollständig geladen 1161
 Image() 454, 684, 707
 image/gif 1556
 image/jpeg 369, 1169, 1170
 image/png 1556
 Image-Control 702, 1722
 Image-Dateiarten 671
 Image-MAP 671
 Image-Map client-seitig 682, 686, 754, 1726
 Image-Map server-seitig 681, 1641
 Image-Medien 702
 IMAGETOOLBAR 749
 IME 811, 820, 1933
 IMG 1095, 1096, 1108, 1255
 ÎMG 1377
 IMG mit Sound 681, 703
 img Objekt 622, 670, 685, 703, 738, 812, 822, 823, 1938
 img Objekt Events 1135, 1956
 img Objekt Filter 551
 img Objekt Styles 835
 IMMEDIATEEND 908, 914, 921, 927, 945, 971, 1635
 IMPLEMENTATION 506, 507
 implementation Objekt 427, 1635
 implementierten Eigenschaften 223
 implementierten Methoden 223
 import 102
 IMPORT 869, 876, 877
 Import Behavior 507, 1761
 Import Verhalten 507, 1761
 in 76, 1657
 in das Clipboard verschieben 277, 476
 in das Dokument ausgeben 447, 1913
 in die Windows Zwischenablage verschieben 277, 476
 in Fenster laden 396
 in Javascript vordefinierte Operationen mit Objekten 142
 in Operator 91
 inaktives Fenster 981
 Indent 277, 476
 Index der Option im select Objekt 760, 1687
 Index des Elementes in der Tab-Tasten-Folge 482, 562, 568, 573, 598, 610, 619, 628, 649, 660, 667, 682, 687, 692, 697, 704, 709, 714, 718, 722, 727, 731, 745, 754, 776, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708
 Index des Elementes in der Tab-Tasten-Folge Body 590, 615, 760
 Index des Objektes in der Collection document.all 482, 598, 602, 610, 628, 649, 660, 663, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 736, 740, 745, 750, 754, 775, 1000, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1188, 1692
 Index in der Collection document.all Body 590, 615, 760
 indexOf() 217, 1828
 indizierte Datenspeicherung 1427
 Infinity 54, 57, 78
 -Infinity 54
 -Infinity 57
 -Infinity 78
 Informationen Betriebssystem 1167
 Informationen Browser 1167
 Informationen CPU 1167
 Informationen über den Browser 410
 Informationen User 1167
 Inhalt des HTML-Elementes 992

Inhalt eines Objektes ausrichten 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1727
 Inhalt Textbereich ersetzen 460, 827, 1084, 1853
 Inhalt Textbereich füllen 460, 827, 1084, 1853
 Inhalt Textbereich leeren 460, 827, 1084, 1853
 inherit 329, 1597
 Initialisierung Filter 518, 2008
 Inkonsistenz 259, 484, 564, 569, 575, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 762, 768, 777, 1025, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1196, 1848
 inline-frame 277, 476
 Inline-Style 482, 562, 568, 573, 598, 610, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 740, 745, 754, 775, 797, 847, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1703
 Inline-Style Body 590, 615, 760, 766
 innerHeight 439
 innerWidth 439
 Input 1260
 INPUT 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1108, 1721
 INPUT BUTTON 1647
 Input button 1261
 input button Objekt 686, 1722
 input button Objekt Events 1135, 1957
 input button Objekt Filter 551
 input button Objekt Styles 835
 Input checkbox 1264
 input checkbox Objekt 248, 690, 1078, 1607, 1699, 1722, 1727
 input checkbox Objekt Events 1136, 1957
 input checkbox Objekt Filter 551
 input checkbox Objekt Grauzustand (Dimmed) und Selektiertheit 691, 1635
 input checkbox Objekt Selektionsstatus 691, 712, 713, 1591, 1606
 input checkbox Objekt Style 690
 input checkbox Objekt Styles 836
 Input file 1268
 input file Objekt 248, 695, 1607, 1722, 1727
 input file Objekt Events 1136, 1957
 input file Objekt Filter 551
 input file Objekt Styles 836
 INPUT HIDDEN 1647
 input hidden Objekt 248, 699, 1607, 1722, 1727
 input hidden Objekt Events 1136, 1957
 input hidden Objekt Filter 552
 input hidden Objekt Styles 837
 input image Objekt 671, 702, 722, 1722
 input image Objekt Events 1136, 1957
 input image Objekt Filter 552
 input image Objekt Styles 837
 Input Method Editor 811, 820, 1933
 input Objekt 248, 593, 622, 632, 685, 1607, 1727
 input Objekt Events 1135, 1957
 input Objekt Filter 551
 input Objekt Focus 690, 694, 699, 707, 711, 716, 720, 725, 729, 1883
 input XE "Objekt input im Formular markieren" XE "Objekt input markieren" XE "input Objekt markieren" Objekt im Formular markieren 690, 694, 699, 706, 711, 716, 720, 725, 729, 1883



Objekt markieren 690, 694, 699, 706, 711, 716, 720, 725, 729, 1883
 Objekt Styles 835
 Objekt Typ 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 Objekt Variante 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 PASSWORD 1647
 password Objekt 248, 707, 1607, 1722, 1727
 password Objekt Events 1136, 1958
 password Objekt Filter 552
 password Objekt Styles 837
☐ radio 1272
☐ radio Objekt 248, 712, 1607, 1722, 1727
☐ radio Objekt Events 1137, 1958
☐ radio Objekt Filter 552
☐ radio Objekt Selektionsstatus 691, 712, 713, 1591, 1606
☐ radio Objekt Styles 838
 reset 1721
 RESET 1647
 reset Objekt 248, 716, 1607, 1722, 1727
 reset Objekt Events 1137, 1958
 reset Objekt Filter 552
 reset Objekt Styles 838
 submit 1721
 SUBMIT 1647
 submit Objekt 248, 721, 1607, 1722, 1727
 submit Objekt Events 1137, 1958
 submit Objekt Filter 553
 submit Objekt Styles 839
 text 1276
 TEXT 1647
 text Objekt 248, 457, 458, 725, 770, 771, 1082, 1607, 1722, 1727, 1883
 text Objekt Events 1137, 1959
 text Objekt Filter 553
 text Objekt Styles 839
☐ type=checkbox 248, 1607, 1722, 1727
 Input-Button-Control 277, 476
 Input-Control 277, 434, 457, 459, 476, 771, 1084, 1768
 INPUT-Element mit Sound 681, 703
 Input-Elemente und Label 729
 Input-Password-Control 277, 477
 Input-Radio-Control 277, 477
 Input-Reset-Control 277, 477
 Input-TextArea-Control 278, 477
 Input-Text-Control 277, 477
 INPUT-Varianten 686
 InsertButton 277, 476
 InsertFieldset 277, 476
 InsertHorizontalRule 277, 476
 InsertIFrame 277, 476
 InsertImage 277, 476
 InsertInputButton 277, 476
 InsertInputCheckbox 277, 476
 InsertInputFileUpload 277, 476
 InsertInputHidden 277, 476
 InsertInputImage 277, 476
 InsertInputPassword 277, 476
 InsertInputRadio 277, 477
 InsertInputReset 277, 477
 InsertInputSubmit 277, 477
 InsertInputText 277, 477
 InsertMarquee 277, 477
 InsertOrderedList 277, 477

InsertParagraph 277, 477
 InsertSelectDropdown 277, 477
 InsertSelectListbox 278, 477
 InsertTextArea 278, 477
 InsertUnorderedList 278, 477
 inset 983, 987, 991
 Inset Filter 533
 instanceof Operator 78, 81
 Instanz 65, 143
 Instanz eines JScript-Objektes 156, 212, 1830
 Instanz ohne Datentyp 54
 instanzieren Dokument 437, 1849
 instanzisiertes Objekt 69
 Instanziierung einer Variablen 64
 Int 1376
 Intel 411, 1601
 interactive 246, 428, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1195, 1679
 Interaktion mit User 866
 Interaktionsfähigkeit Objekt 481, 482, 561, 567, 572, 582, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 686, 687, 691, 696, 700, 703, 708, 713, 717, 721, 725, 726, 730, 736, 739, 743, 744, 749, 750, 751, 753, 759, 765, 766, 774, 775, 1016, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1610, 1640
 interne Zeigerverwendung 329
 interner Wert document.select.option Objekt 767, 1711
 interner Wert document.select.option Objekt 767
 interner Wert document.select.option Objekt 1711
 interner Wert Objekt document.select.option 767, 1711
 interner Wert Option 767, 1711
 internes ID 481, 561, 567, 572, 597, 601, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 Internet 974
 Internet Explorer 1104, 1142, 1147, 1150, 1152, 1160, 1161, 1167
 Internet Explorer 6.x-Inkompatibilitätsmodus zu seinen Vorgängern 802
 Internet Explorer 6.x-Kompatibilitätsmodus zu seinen Vorgängern 802
 Internet Explorer bedingtes Parsen 53, 99
 Internet Explorer Behavior 850
 Internet Explorer Browsererkennung 38
 Internet Explorer Browsererweiterungen 1375
 Internet Explorer Browser-Erweiterungen 749
 Internet Explorer CSS und Attribute width und height 803
 Internet Explorer CSS1- Inkonformität 801
 Internet Explorer CSS1- Kompatibilität auch bei Frameset-Dokument 802
 Internet Explorer CSS1- Kompatibilität außer in einem Frameset-Dokument 802
 Internet Explorer CSS1- Konformität genau im Umfang von HTML 4 laut World Wide Web Consortium 802
 Internet Explorer CSS1-Konformität 801
 Internet Explorer CSS-Konformität der Versionen 801
 Internet Explorer Datenbank 1376
 Internet Explorer Direct Animation 1382
 Internet Explorer Fehlermeldungen 66, 114
 Internet Explorer Filter 810, 820, 1931
 Internet Explorer IE 6.x und CSS 424, 1595
 Internet Explorer IE 6.x und CSS1 424, 1595
 Internet Explorer Parsen bedingt 53, 99



Internet Explorer Pars-Reihenfolge 605, 774, 807
 Internet Explorer Standard-Behavior 850
 Internet Explorer Style-Eigenschaften, die nur im HEAD des Dokumentes ansprechbar sind 818
 Internet Explorer Style-Eigenschaften, die nur per Script ansprechbar sind 816
 Internet Explorer Style-Eigenschaften, die nur per STYLE-Attribut ansprechbar sind 818
 Internet Explorer Style-Eigenschaftenübersicht zu Script- und STYLE-Kodierung 818
 Internet Explorer Time-Formate 872
 Internet Explorer und Filter 508
 Internet Explorer und seine Scriptmaschine 18, 31, 42, 135
 Internet Media Type 753, 1594
 Internet Optionen-Sicherheit 665
 Internet-Provider 24
 InternetShortcut 862
 Intern Explorer 1145
 Interpretation des HTML-Dokumentes 40
 Interpretation des Javascript-Codes 40
 Interpretation einer Javascript-Funktion 40
 interpretieren 222
 Intranet 974
 Invert Filter 534
 IP 561, 572, 1167, 1633
 irgendeine Maustaste drücken 1101
 irgendeine Maustaste loslassen 1102
 Iris Filter 534
 irisWipe 1704, 1723
 ISMAP 325, 670
 ISO-Latin-1-Zeichensatz 217, 1775
 isPrototypeOf() 150, 209, 1677
 italic 987, 989
 Italic 278, 477
 Java nicht abgeschaltet 1168
 Java-Applet 450, 566, 570
 Javacode Ausführbarkeit 1170, 1834
 javaEnabled() 1168
 Java-Maschine 1167
 Java-Maschine im Browser 1170, 1834
 javascript 1649
 Javascript 1187
 JavaScript 30, 39, 136, 402, 404, 1723
 Javascript 1.1 148
 Javascript 1.2 148
 Javascript 1.5 Anweisungen 101
 Javascript 1.5 Operatoren 83
 Javascript Anweisungen 85
 Javascript Datei 46
 Javascript Dialekt 36
 Javascript Fehlerbehandlung 114
 Javascript in HTML einbinden 40
 Javascript reservierter Bezeichner 52
 Javascript und Browserperformance 48
 Javascript und Objekte 131
 Javascript vordefinierte Objekte 160
 javascript: 41
 javascript:void(0); 77
 JavaScript1.1 30, 39
 JavaScript1.2 30, 39
 Javascript-Ausdruck 41
 Javascriptcode ausführen 40
 Javascript-Code Interpretation 40
 Javascript-Code mehrfach verwenden 45
 Javascript-Datei *.js einbinden 45

Javascript-Dialekte 17
 Javascript-Direktkodierung in das HTML-Dokument 40
 Javascript-Elemente 52
 Javascript-Entity 41
 Javascript-Funktionen ausführen 40
 Javascript-Interpreter 52
 Javascript-Kodierung als Aktion eines Eventhandlers 41
 Javascript-Kodierung als Wert eines HTML-Attributes 41
 Javascript-Kodierung in externer Datei 45
 Javascript-Kodierung per HTML-Tag 40
 Javascript-Kodierung zur Laufzeit des HTML-Dokumentes 42
 Javascript-Kommentar 53
 Javascript-Version Erkennung 39
 Javascript-Versionen 17
 Javascript-Versionen bei Microsoft 30
 Javascript-Versionen beim Netscape 30
 Java-Verfügbarkeit 412, 1834
 Javascript Ausführung 425, 1607
 Javascript-Dateien auf dem Server 48
 jjjj-mo-ddThh:mm:ss+hh:mm.TZD 872
 Joint Photographic Experts Group 671, 702
 JPEG 671, 702
 JPG 671, 702, 983, 988
 Jscript 30, 40, 1187
 JScript 402, 404, 1649, 1723
 JScript Anweisungen 85, 99
 JScript Fehlerbedingung 98, 129
 JScript Fehlerbehandlung 97, 128
 JScript Kommentar 85
 JScript Laufzeit-Bibliothek 1420
 JScript Objekt Array VisualBasic-Array nach JScript-Array konvertieren 171
 JScript Runtime Fehler 115
 JScript Runtime Fehler 5000 115
 JScript Runtime Fehler 5001 115
 JScript Runtime Fehler 5002 115
 JScript Runtime Fehler 5003 115
 JScript Runtime Fehler 5005 115
 JScript Runtime Fehler 5006 115
 JScript Runtime Fehler 5007 115
 JScript Runtime Fehler 5008 115
 JScript Runtime Fehler 5009 115
 JScript Runtime Fehler 5010 115
 JScript Runtime Fehler 5012 115
 JScript Runtime Fehler 5013 115
 JScript Runtime Fehler 5014 115
 JScript Runtime Fehler 5015 115
 JScript Runtime Fehler 5016 115
 JScript Runtime Fehler 5017 115
 JScript Runtime Fehler 5018 115
 JScript Runtime Fehler 5019 115
 JScript Runtime Fehler 5020 115
 JScript Runtime Fehler 5021 115
 JScript Runtime Fehler 5022 115
 JScript Runtime Fehler 5023 115
 JScript Runtime Fehler 5024 115
 JScript Runtime Fehler 5025 115
 JScript Runtime Fehler 5026 115
 JScript Runtime Fehler 5027 115
 JScript Runtime Fehler 5028 115
 JScript Runtime Fehler 5029 115
 JScript Runtime Fehler 5030 115
 JScript Scriptmaschine Buildnummer 40, 1875
 JScript Scriptmaschine Hauptversion 40, 1875



JScript Scriptmaschine Sprache 40, 1875
 JScript Scriptmaschine Unterversion 40, 1876
 JScript Spezialzeichen 113
 JScript Standard-Eigenschaften und -Methoden aller Objekte 149
 JScript Standard-Methoden aller Objekte 149
 JScript Syntax Fehler 115
 JScript Syntax Fehler 1002 128
 JScript Syntax Fehler 1003 128
 JScript Syntax Fehler 1004 128
 JScript Syntax Fehler 1005 128
 JScript Syntax Fehler 1006 128
 JScript Syntax Fehler 1007 128
 JScript Syntax Fehler 1008 128
 JScript Syntax Fehler 1009 128
 JScript Syntax Fehler 1010 128
 JScript Syntax Fehler 1011 128
 JScript Syntax Fehler 1012 128
 JScript Syntax Fehler 1014 128
 JScript Syntax Fehler 1015 128
 JScript Syntax Fehler 1016 128
 JScript Syntax Fehler 1018 128
 JScript Syntax Fehler 1019 128
 JScript Syntax Fehler 1020 128
 JScript Syntax Fehler 1023 128
 JScript Syntax Fehler 1024 128
 JScript Syntax Fehler 1025 128
 JScript Syntax Fehler 1026 128
 JScript Syntax Fehler 1027 128
 JScript Syntax Fehler 1028 128
 JScript Syntax Fehler 1029 128
 JScript Syntax Fehler 1030 128
 JScript Syntax Fehler 1031 128
 JScript Syntax Fehler 1032 128
 JScript Syntax Fehler 1033 128
 JScript Syntax Fehler 1035 128
 JScript und Browserperformance 48
 JScript und Dateisystem 1430
 JScript und seine Operatoren 77
 JScript unter Windows 98 18, 31, 42, 135
 JScript unter Windows XP 18, 31, 42, 135
 JScript.NET 224
 JScript-Maschine Erkennung 40
 JScript-Objekt 149, 208
 JScript-Objekt arguments 149
 JScript-Objekt Array 170
 JScript-Objekt Array 1855
 JScript-Objekt Array 1859
 JScript-Objekt Array 1899
 JScript-Objekt Array 1912
 JScript-Objekt Array Elemente anhängen 1859
 JScript-Objekt Array erstes Feldelement liefern und löschen 1899
 JScript-Objekt Array letztes Feldelement liefern und löschen 1855
 JScript-Objekt Array Werte dem Feld voransetzen 1912
 JScript-Objekt Bezeichner 61, 150, 209, 1595, 1596
 JScript-Objekt Boolean 149
 JScript-Objekt Date 149
 JScript-Objekt Eigenschaft prüfen 155, 211, 1825
 JScript-Objekt Enumerator 149, 188, 1740, 1831, 1840
 JScript-Objekt error 192, 1607, 1656, 1659, 1662
 JScript-Objekt Error 149
 JScript-Objekt Function 149
 JScript-Objekt Instanz prüfen 156, 212, 1830

JScript-Objekt Math 149
 JScript-Objekt Methode prüfen 155, 211, 1825
 JScript-Objekt Number 149
 JScript-Objekt Object 62, 149, 208
 JScript-Objekt Prototyping 150, 209, 1677
 JScript-Objekt RegExp 149
 JScript-Objekt String 149
 JScript-Objekt var 149
 JScript-Objekt Wert ermitteln 159, 213, 1913
 JScript-Objektklasse Bezeichner 61, 150, 209, 1595, 1596
 Jscript-Scriptmaschine Buildnummer 100
 Jscript-Scriptmaschine Version 100
 JS-Datei 46
 justify 986, 990, 1562
 JustifyCenter 278, 477
 JustifyLeft 278, 477
 JustifyRight 278, 477
 kacheln 327, 983
 kacheln Hintergrundbild 809, 819, 1919
 Kapitälchen 987, 989
 Kapselung von Daten 44, 72
 Kapselung von Methoden 44, 72
 Kashida 815, 825, 1946
 keine Maustaste gedrückt 1104
 KEINE Suchmaschine darf scannen 49
 Kette anhängen 281, 602, 1739
 Kettenlänge 58
 Kettenoperationen 43, 214
 keydown 1152
 keypress 1152
 keyup 1153
 KeyUp 1153, 1155
 Keywords 748
 Kind 222, 250, 482, 563, 568, 573, 591, 599, 602, 610, 616, 620, 629, 649, 660, 663, 668, 682, 688, 693, 697, 701, 705, 710, 714, 719, 723, 727, 732, 737, 740, 745, 754, 761, 767, 776, 1019, 1029, 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1189, 1192, 1739
 Kind Abstand zum Elternobjekt 666, 681, 1634
 Kind als Knoten entfernen 241
 Kind als Knoten ersetzen 241
 Kind als Knoten erzeugen 241
 Kind anhängen 250, 591, 599, 602, 610, 616, 619, 628, 660, 663, 667, 688, 693, 697, 705, 709, 714, 718, 723, 727, 732, 740, 745, 767, 776, 1019, 1029, 1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1192, 1739
 Kind erstes 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 Kind ERSTES 241
 Kind letztes 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Kind LETZTES 241
 Kind NACHFOLGENDES 241
 Kind nachfolgendes 243, 561, 567, 572, 590, 598, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1660



Kind Name 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661

Kind Objekt 260, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 741, 747, 762, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1194, 1867

Kind Vorgänger 246, 562, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 760, 766, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1077, 1192, 1675

Kind VORHERGEHENDES 241

Kindeigenschaft 240

Kinder des Objektes 268, 271, 467, 471

Kinder Existenz 258, 564, 569, 574, 592, 600, 602, 611, 617, 621, 630, 650, 661, 664, 668, 683, 689, 694, 698, 706, 710, 715, 719, 724, 728, 733, 737, 741, 746, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1189, 1193, 1824

Kind-Existenz 240

Kind-Existenz möglich 242, 481, 560, 597, 601, 614, 618, 627, 659, 662, 666, 686, 691, 695, 703, 708, 712, 717, 721, 725, 730, 739, 743, 752, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1069, 1076, 1191, 1589

Kind-Fenster 221, 382, 1666

Kindknoten 259, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 698, 706, 711, 715, 720, 724, 728, 733, 741, 747, 755, 762, 768, 777, 1021, 1031, 1035, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1194, 1829

Kind-Name 241

Klassenbezeichner ActiveX Control 752, 1591

Klassenname 560, 567, 571, 627, 648, 659, 666, 681, 686, 691, 696, 700, 703, 708, 712, 717, 721, 725, 730, 739, 743, 753, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1191, 1591

Klassenreferenz 560, 567, 571, 609, 618, 627, 648, 659, 666, 681, 686, 691, 696, 700, 703, 708, 712, 717, 721, 725, 730, 739, 743, 753, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1191, 1591

Klassenreferenz Body 589, 614, 759, 765

klein 987, 989

Klein oder Gross 1101

Kleinbuchstabe 987, 990

kleinere der beiden Zahlen 199, 1839

Klick auf das Element 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 683, 688, 693, 698, 705, 710, 715, 719, 723, 727, 732, 740, 746, 754, 761, 767, 776, 1019, 1030, 1043, 1048, 1054, 1060, 1066, 1072, 1079, 1193, 1748

Klick auf Scrollbar simulieren 483, 1079, 1763

klonen 239

klonen Objekt 251, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1749

Knoten 222

Knoten als Kind anhängen 250, 591, 599, 602, 610, 616, 619, 628, 660, 663, 667, 688, 693, 697, 705, 709, 714, 718, 723, 727, 732, 740, 745, 767, 776, 1019, 1029,

1034, 1037, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1192, 1739

Knoten als Kind entfernen 241

Knoten als Kind ersetzen 241

Knoten als Kind erzeugen 241

Knoten Attribut 263, 484, 565, 570, 575, 593, 601, 603, 612, 618, 622, 631, 651, 662, 665, 669, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742, 747, 751, 752, 756, 763, 769, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068, 1074, 1082, 1190, 1194, 1196, 1887

Knoten DOM-Position 240

Knoten Eltern 240, 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668

Knoten Elternobjekt 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1667

Knoten entfernen 240, 261, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 741, 747, 756, 763, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1194, 1868

Knoten ersetzen 240

Knoten Existenz 258, 564, 569, 574, 592, 600, 602, 611, 617, 621, 630, 650, 661, 664, 668, 683, 689, 694, 698, 706, 710, 715, 719, 724, 728, 733, 737, 741, 746, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1189, 1193, 1824

Knoten im DOM tauschen 266, 565, 570, 575, 593, 601, 603, 613, 618, 622, 632, 651, 662, 665, 670, 684, 690, 695, 699, 702, 707, 712, 716, 721, 725, 729, 734, 738, 742, 748, 756, 763, 769, 778, 1027, 1032, 1035, 1039, 1045, 1051, 1056, 1062, 1068, 1074, 1082, 1190, 1195, 1909

Knoten Kind 259, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 698, 706, 711, 715, 720, 724, 728, 733, 741, 747, 755, 762, 768, 777, 1021, 1031, 1035, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1194, 1829

Knoten tauschen in der DOM-Position 240

Knoten und seine Attribute 238

Knotentyp 238, 239, 244, 561, 567, 572, 598, 601, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661

Knotentyp Body 590, 614, 760, 766

Knotenwert 238, 239, 244, 561, 567, 572, 598, 601, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 704, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661

Knotenwert Body 590, 614, 760, 766

kodieren von Literal in das Unicode-Format 144, 153, 211, 1766

kodieren von String oder Literal in das Unicode-Format 144, 153, 211, 1766

Kodierung von CSS-Werten in Style-Sheets 805

Kodierung von Werten mit erlaubter Einheit 805

Kodierungsregeln 52

Kombination Steuertaste und andere Taste 1100



Kombination von Literalen, Variablen 67
 Komma 75, 980
 Komma für Dezimal komma 1376
 Kommando Ausführbarkeit 279, 441, 457, 460, 479, 771, 1085, 1859
 Kommando ausführen 276, 434, 457, 459, 476, 771, 1084, 1768
 Kommando Ausführungserfolg 279, 441, 457, 460, 479, 771, 1085, 1859
 Kommando Unterstützung 280, 441, 457, 460, 479, 771, 1085, 1859
 Kommando vordefiniert 275, 474
 Kommando Wert 280, 479
 Kommandos Wert 441, 457, 460, 771, 1085, 1859
 Kommando-Status 279, 441, 457, 460, 479, 771, 1085, 1859
 Kommentar 53, 252, 432, 601, 665, 1754
 Kommentar Anzahl der Zeichen 601, 1650
 Kommentar einzeilig 85
 Kommentar in einer Tabelle 1018, 1704
 Kommentar in JScript 85
 Kommentar mehrzeilig 85
 Kommentare innerhalb <STYLE> <STYLE> 804
 Kommentar-Objekt 238
 Kommunikation von Dokumenten 426, 1616
 Kompatibilität des IE 6.x zu CSS1 424, 1595
 Komplementärfarben Filter 512, 2002
 komplettes Laden 1096
 Komponenten der Variablen 62
 Komponenten des Betriebssystems 1142
 konsistenten Struktur 259, 484, 564, 569, 574, 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 738, 741, 747, 751, 755, 762, 768, 777, 1025, 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073, 1081, 1190, 1194, 1196, 1848
 Konsistenz DOM 222
 Konstruktor 111, 164
 Konstruktor privat 61, 150, 209, 1595, 1596
 Kontextänderung eines Objektes 1980
 Kontextmenü 1097, 1098, 1102, 1115, 1162, 1981
 Kontextmenü im Fenster 1162, 1656
 Kontextmenü-Eintrag erzeugen 1457
 Kontextmeü 1098, 1103
 konvertieren zu ganzzahlig 197, 203
 konvertieren zu ganzzahlig teilbar 198, 203
 Konvertierung einer Objektinstanz in eine 32-Bit-Zahl 149
 Konvertierung numerisch 200
 Konvertierung Wert nach Boolean 152, 1745
 -Koordinate der linken oberen Ecke Objektes 459, 482, 561, 567, 572, 598, 610, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1083, 1191, 1663
 Koordinate des Ausschnittes links 849, 1593
 Koordinate des Ausschnittes oben 849, 1594
 Koordinate des Ausschnittes rechts 849, 1594
 Koordinate des Ausschnittes unten 849, 1593
 Kopf der Funktion 88, 106
 Kopfnote 983, 998, 1553, 1678
 Kopfzeile des Dokumentes 419
 Kopieraktion 1098
 Koten als Kind 238
 Koten und sein Eltern-Objekt 238
 Kreisform 1688

Kreuz mit Beweglichkeit 988
 kursiv 987, 989
 Kurztastenkombination 809, 818, 1918
 Label 721
 label Anweisung 85, 86, 92
 Label des Elementes 248, 1607, 1722, 1727
 Label des Reset-Button 716
 Label document.select.option Objekt 766, 1649
 Label einer Option 766, 1649
 Label ID 730, 1634
 Label Input-Elemente 729
 Label mit Tastenkombination 1918
 label Objekt 248, 729, 1607, 1722, 1727
 Label Objekt document.select.option 766, 1649
 label Objekt Events 1138, 1959
 label Objekt Styles 839
 Lade-Ereignisse des Internet Explorer ab 4.x bzw. 5.x 1161
 Lade-Ereignisse des Netscape ab 3.x 1161
 Lade-Ereignisse für Bild 1161
 Lade-Ereignisse für HTML-Dokument 1162
 Lade-Eventbehandlung 1161
 Laden 1096
 Laden Abbruch 1095
 laden Bild 1161
 laden den Dokumentes 222
 Laden des HTML-Dokumentes 70
 laden eines Bildes 129, 1100, 1147
 Laden eines fremden Dokumentes ohne Framedarstellung 313, 319, 643, 653
 laden HTML-Dokument 395, 1844
 laden HTML-Dokument mit all seinen Elementen 1101
 laden Image 1161
 laden in das Fenster 396
 laden neu des aktuellen Dokumentes 1167, 1865
 laden neues Dokument 1167, 1739, 1869
 laden Objekt 567, 752, 1565
 ladens eines HTML-Dokumentes 129, 1100, 1147
 Ladevorgang 1161
 Ladevorgang eines Bildes 1097
 Ladezustand Objekt 681, 686, 703, 1595
 Lage auf dem Server 49
 Lage der Cookies 486
 Lage der Homepage 857, 1830
 Lage der Objekte im Quellcode 222
 Ländereinstellung 158, 212, 1910
 LANG 748
 Länge Textbox 708, 726, 1654
 Länge Text-Control 708, 726, 1654
 Länge Textfeld 708, 726, 1654
 LANGUAGE 1187
 large 987, 989
 larger 987, 989
 laufende Nummer der Option in List-Box 765, 1635
 Laufschrift in einem Formular 640
 Laufwerk 1432, 1764, 1785
 Laufwerk Bereitschaft 1434, 1646
 Laufwerk freier Speicher in Bytes 1434, 1574, 1627
 Laufwerk gesamter Speicher in Bytes 1435, 1721
 Laufwerk Netzwerkname 1435, 1688
 Laufwerk Seriennummer 1435, 1687
 Laufwerk Volumenbezeichner 1435, 1733
 Laufwerksbuchstabe 1432, 1433, 1434, 1617, 1785, 1786
 Laufwerkstyp 1434, 1617
 Laufwerkszugriff 1434



Laufzeitfehler 128
 Laufzeitfehler abfangen 128
 Lautstärke Widergabe 583, 1731
 Lautstärkeeinstellung von Windows 576
 Layer 1101, 1105
 LAYER 326, 980
 Layer dynamisch erzeugen 311, 331
 Layer Eigenschaften 311, 332
 Layer gemeinsam übergeordnet 329
 Layer Hintergrundfarbe 328
 Layer Inhalt 311, 332
 Layer logischer Name 328
 Layer nächst tieferer 328
 Layer nächster übergeordneter 327
 layer Objekt 604, 772, 816, 826, 1950
 Layer Sichtbarkeit 329
 Layer transparenter Hintergrund 328
 Layer und sein HTML-Dokument 329
 Layer unverschachtelt 310
 Layer verschachtelt 310, 331
 Layer verschieben 311, 332
 Layer-Element 985
 Layerfolge 329
 Layer-Objekt 772
 Layerverschachtelung 329
 Layout 481, 482, 560, 561, 567, 572, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 686, 687, 691, 696, 700, 703, 708, 713, 717, 721, 725, 726, 730, 736, 739, 743, 744, 750, 751, 753, 774, 775, 1076, 1077, 1188, 1191, 1195, 1596, 1640
 Layout des Dokumentes 222, 419
 Layout per Style 797
 Layout Tabelle 812, 815, 822, 824, 1937, 1944
 Layout-Komponente eines Objektes 483, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 750, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751
 Leeranweisung 85
 leere Argumentenliste 40
 Leerkette 58
 Leertaste 1101
 Leerzeichen 56, 57, 58, 113, 114, 144
 left 397, 439, 483, 986, 990, 1080, 1562, 1610, 1763
 left Margin Body 590, 1650
 LEFTMARGIN 584
 leftToRight 1703, 1723
 Leisten 439
 length 636, 1165
 Leserichtung 816, 818, 825, 1947
 LETZTES Kind 241
 letztes Kind Zeiger 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 letztes Update Datum 681, 1626
 lhs 1016, 1627
 Light Filter 535
 lighter 987, 989
 line 811, 821, 1933
 Line feed 56, 58, 114
 Line Feed 56, 58, 113, 114
 linear 1588
 lineare Transformation 512, 2001

line-through 987, 990
 Linie dynamisch erzeugt 325, 670
 Linie horizontal 277, 325, 476, 670
 Linie mit variabler Dimension 325, 670
 Linie mit variabler Höhe 325, 670
 Linie mit variabler Länge 325, 670
 Linie vertikal 325, 670
 link 1617
 Link 278, 382, 428, 477, 1101, 1659, 1679, 1722
 LINK 1280
 Link aktiv 809, 818, 1917
 Link der bereits aktiviert wurde 809, 818, 1917
 Link der nicht kürzlich aktiviert wurde 809, 818, 1917
 Link erzeugen 277, 476
 Link Farbe 590, 1651
 Link Klick-Wirkung 572, 1662
 Link mit Maus überfahren 809, 818, 1917
 link Objekt 559, 734
 link Objekt Events 1138, 1959
 link Objekt Styles 840
 Link ohne Vverweiswert 77
 Link per HREF 1096
 linke ALT-Taste 1104
 linke Ctrl-Taste 1104
 linke Maustaste 1102, 1106
 linke Maustaste gedrückt 1104
 linke obere Ecke 993
 linke obere Ecke des HTML-Dokumentes 1104
 linke obere Ecke des Objektes 459, 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1083, 1191, 1664
 linke oberen Ecke Objektes 459, 482, 561, 567, 572, 598, 610, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1083, 1191, 1663
 linke oder rechte Maustaste 1101
 linke Strg-Taste 1104
 linker Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 linker Rand des Objektes 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 714, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 linkMove 1620
 Links 455, 738, 984
 Links aktiv Farbe 424, 1564
 Links bereits geklickt Farbe 430, 1731
 Links nicht benutzt Farbe 427, 1651
 linksbündig 986, 990
 Linkt mit Textangabe 559
 List-Box 757
 List-Box Anzahl der Zeilen 760, 1692
 List-Box laufende Nummer der Option 765, 1635
 List-Box Option laufende Nummer 765, 1635
 List-Box Zeilenanzahl 760, 1692
 List-Box-Selektion 278, 477
 Liste 277, 477, 985
 Liste von Multipurpose Internet Mail Extensions (MIME)-Typen 1555



Listendarstellung Aufzählungsliste-Elemente 811, 821, 1934
 Listendarstellung Marker als Bild 811, 821, 1935
 Listendarstellung Marker als Image 811, 821, 1935
 Listendarstellung Marker Position 811, 821, 1935
 Listendarstellung Markertyp (nicht Image) 811, 821, 1935
 Literal 56, 213
 Literal Basis-Datentyp 56
 Literal Datentyp 56
 Literal im Unicode-Format dekodieren 213
 Literal in das Unicode-Format kodieren 144, 153, 211, 1766
 Literal numerisch 205
 LN10 199, 1651
 LN2 199, 1651
 loaded 246, 428, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1195, 1679
 loading 246, 428, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1195, 1679
 Local Machine 974
 localized-text 1604
 location 397, 427, 439, 1281, 1651
 location Objekt 281, 283, 370, 382, 419, 1166
 location.hostname 25
 location.href 395, 1844
 location-Objekt 1651
 locked 911, 917, 924, 930, 934, 949, 957, 974, 1131, 1705, 1999
 Log von 2 zur Basis 10 199, 1652
 Log von e zur Basis 2 199, 1652
 Log zur Basis 10 199, 1651
 Log zur Basis 2 199, 1651
 LOG10E 199, 1652
 LOG2E 199, 1652
 Logarithmus zur Basis E 199, 1838
 logische Objekthierarchie 222
 logische Operationen 75
 logische Operatoren 75
 logischer Name 225
 logischer Name des Layers 328
 logischer Objektname 224, 481, 561, 567, 572, 597, 601, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 logischer Window-Name 437
 logischer Windowsname 440
 Logox Webspeech 1498
 LOGOX-Datei 1509
 lokale Gültigkeit 65
 lokale Pfadangabe 56, 58, 114
 lokales Programm starten 1452
 LONGTRANSITION 908, 914, 921, 927, 945, 971, 1652
 loop 748, 1481, 1907
 Loop 890, 893, 898, 904, 909, 915, 922, 927, 932, 945, 954, 965, 971, 1679
 LOOP 576, 581, 1115, 1119, 1130, 1980, 1985, 1998
 löschen Cookie 489
 löschen des Formulars 637
 Löschen einer Objektinstanz 143
 löschen Selektion 770, 1764
 Löschen von per AutoComplete gespeicherter Werte 1573
 Löschung der Variable 65, 143

Löschung einer Variable 65
 loslassen der linken Maustaste 1102
 loslassen irgendeiner Maustaste 1102
 LOWER 941
 lowercase 987, 990
 lowsrc 671
 LOWSRC 325, 670, 671
 ltr 481, 849, 1581, 1610
 Mac68K 412, 1672
 Machcode 114
 Macintosh 68K 1672
 Macintosh PowerPC 1672
 Macintosh Power-PC 412
 MacPPC 412, 1672
 Mailtext einer Email 626, 1559
 mailto 626, 1559
 Majorversion Browser 411, 1567
 MAME 309, 451, 613, 756
 map Objekt 455, 571, 738, 742
 map Objekt Events 1138, 1959
 map Objekt Styles 840
 MAP-Image 671
 Mappe 738
 Margin 589, 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 MARGINHEIGHT 584
 MARGINWIDTH 584
 markieren bzw. entmarkieren des Buttons 637
 markiert den Inhalt des Eingabefeldes 634, 635
 Markierung 1103
 Markierung der Selektion 770, 1746
 Marquee 277, 477
 MARQUEE 743, 1281
 marquee Objekt 742
 marquee Objekt Anzahl der Scrollaktionen 744
 marquee Objekt endlos scrollen 1653
 marquee Objekt Events 1138, 1959
 marquee Objekt Filter 553
 marquee Objekt Scrollaktionen Anzahl 744
 marquee Objekt Scrollaktionen Stop 748, 1907
 marquee Objekt Scrollgeschwindigkeit 744, 1684
 marquee Objekt Scrollrichtung 1610
 marquee Objekt Scrollrichtung beim Start 743, 1610
 marquee Objekt Scrollschrittweite 744, 1684
 marquee Objekt Scroll-Takteinheit 745, 1721
 marquee Objekt Scrollverhalten 743
 marquee Objekt Start Scrollaktion 747, 1906
 marquee Objekt Styles 840
 MaskFilter Filter 535
 Math JScript-Objekt 149
 Math Objekt 149, 160
 Math Script-Objekt 62, 150, 160, 195, 209, 1595, 1596, 1620, 1651, 1652, 1672, 1694, 1734, 1739, 1745, 1753, 1769, 1775, 1838, 1839, 1855, 1860, 1874, 1905, 1906, 1909
 Math.ceil() 76
 Math.floor() 165, 200, 420, 1987, 2000
 Math-Objekt 76
 Matrix 164
 Matrix Filter 515, 536, 2005
 Matrix zweidimensional 164
 Matrixfeld Filter 513, 2002
 Maus Eventobjekt 1107, 1630
 Maus hat das Objekt angeklickt 1098
 Maus im Layer 1105



Maus verlässt den Bereich des Objektes 1102
 Maus X-Koordinate 1107, 1592, 1665, 1684
 Maus Y-Koordinate 1107, 1592, 1665, 1684
 Mausbewegung 1102
 Mauscursor 988
 Mauscursor Art 810, 820, 1930
 Maus-Event auslösendes Objekt 1107, 1720
 Mausklick 1096
 Mausposition 1104, 1158
 Mausposition im HTML-Element 1105
 Mausevent 1107, 1121, 1733, 1988
 Maustaste 1106, 1158, 1160
 Maustaste die das Event auslöst 1107, 1584
 Maustaste drücken 1101
 Maustaste gedrückt 1104
 Maustaste linke 1102
 Maustaste linke oder rechte 1101
 Maustaste links 1158, 1584
 Maustaste loslassen 1102
 Maustaste mitte 1158, 1584
 Maustaste mittlere 1101, 1102
 Maustaste rechte 1102
 Maustaste rechte oder linke 1101
 Maustaste rechts 1158, 1584
 Maustastenkombination 1158
 Mausüberwachung 1120, 1987
 Maus-Überwachung ausschalten für ein Objekt 446, 484, 564, 569, 575, 592, 600, 612, 617, 621, 630, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 741, 747, 755, 762, 768, 777, 1026, 1031, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1107, 1194, 1865
 Maus-Überwachung einschalten für ein Objekt 484, 565, 570, 575, 593, 601, 612, 618, 622, 631, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 742, 747, 756, 763, 769, 778, 1027, 1031, 1045, 1051, 1056, 1062, 1068, 1074, 1082, 1107, 1194, 1887
 Mauszeiger 1096
 MAX_VALUE 204, 1654
 maximal verfügbare Fensterhöhe 1182
 Maximale Anzahl der durch User eingebbaren Zeichen in einem Text-Control 708, 726, 1654
 maximale Bildschirmauflösung 1182
 maximale verfügbare Fensterbreite 1182
 maximaler numerischer endlicher Wert 204
 MAXLENGTH 725
 MAYSCRIPT 306, 450, 566, 570
 Media 907
 MEDIA 981, 982
 Media Bar 396, 437, 860, 1709
 Media Bar aktive Wiedergabe stoppen 859, 1907
 Media Bar aktueller Media-Eintrag 858, 1603
 Media Bar Anzahl der mit dem MediaItem Objekt verbundenen Attribute 860, 861, 1568
 Media Bar Dauer der Media-Datei in Sekunden 860, 1620
 Media Bar Dauer des MediaItem Objektes in Sekunden 860, 1620
 Media Bar Eintrag in der Playliste Nachfolger 859, 1631
 Media Bar Laden und Wiedergabe einer Media-Datei 859, 1855
 Media Bar Ländernummer Media-Datei 859, 1666
 Media Bar Lizenz Media-Datei 859, 1666
 Media Bar Media-Datei aktive Wiedergabe stoppen 859, 1907
 Media Bar Media-Datei Dauer in Sekunden 860, 1620
 Media Bar Media-Datei Ländernummer 859, 1666

Media Bar Media-Datei Lizenz 859, 1666
 Media Bar Media-Datei von einem Server laden 859, 1855
 Media Bar Media-Datei von einem Server wiedergeben 859, 1855
 Media Bar Media-Datei Wiedergabe Statusänderung 860, 1126, 1993
 Media Bar Media-Datei Wiedergabe stoppen 859, 1907
 Media Bar Media-Datei wiedergeben 859, 1855
 Media Bar Media-Eintrag aktuell 858, 1603
 Media Bar MediaItem Objekt Name 860, 861, 1659
 Media Bar MediaItem Objekt Url 860, 1692
 Media Bar Nachfolger Eintrag in der Playliste 859, 1631
 Media Bar nächsten Eintrag in der Playliste wiedergeben 859, 1854
 Media Bar Name des MediaItem Objektes 860, 861, 1659
 Media Bar Name des mit dem MediaItem Objekt verbundenen Attributes 860, 861, 1778
 Media Bar Player 1470, 1474
 Media Bar Player aktive Wiedergabe stoppen 859, 1907
 Media Bar Player anzeigen 860, 1119, 1130, 1986, 1998
 Media Bar Player bereitet gerade eine neue Media-Wiedergabe vor 1675
 Media Bar Player gibt gerade einen Daten-Stream wider 1675
 Media Bar Player hat das Ende der Media-Datei erkannt 1675
 Media Bar Player hat das Ende des letzten Daten-Streams erkannt 1675
 Media Bar Player Media-Datei aktive Wiedergabe stoppen 859, 1907
 Media Bar Player Media-Datei Dauer in Sekunden 860, 1620
 Media Bar Player Media-Datei von einem Server laden 859, 1855
 Media Bar Player Media-Datei von einem Server wiedergeben 859, 1855
 Media Bar Player Media-Datei Wiedergabe Statusänderung 860, 1126, 1993
 Media Bar Player Media-Datei Wiedergabe stoppen 859, 1907
 Media Bar Player Media-Datei wiedergeben 859, 1855
 Media Bar Player nächsten Eintrag in der Playliste wiedergeben 859, 1854
 Media Bar Player puffert gerade Media-Daten 1675
 Media Bar Player Sichtbarkeit 859, 1621
 Media Bar Player springt geraden den Nachfolger-Daten-Stream an 1675
 Media Bar Player springt geraden den Vorgänger-Daten-Stream an 1675
 Media Bar Player Status des bezüglich Codec 859, 1666
 Media Bar Player Status des bezüglich Individualisierung 859, 1666
 Media Bar Player Status des bezüglich Lizenz 859, 1666
 Media Bar Player Status des bezüglich Playlist 859, 1666
 Media Bar Player Statusänderung bezüglich Codec 860, 1124, 1991
 Media Bar Player Statusänderung bezüglich Individualisierung 860, 1124, 1991
 Media Bar Player Statusänderung bezüglich Lizenz 860, 1124, 1991
 Media Bar Player Statusänderung bezüglich Playliste 860, 1124, 1991



Media Bar Player Statusänderung bezüglich Wiedergabe 860, 1126, 1993
 Media Bar Player wartet gerade auf Daten-Stream 1675
 Media Bar Player Wiedergabe ist gestoppt 1675
 Media Bar Player Wiedergabe Statusänderung 860, 1126, 1993
 Media Bar Player Wiedergabe stoppen 859, 1907
 Media Bar Player Wiedergabestatus 859, 1675
 Media Bar Player Windows Media Player 858, 859, 860, 1119, 1124, 1126, 1130, 1621, 1666, 1675, 1986, 1991, 1993, 1998
 Media Bar Playliste 859, 861, 1666
 Media Bar Playliste aktueller Media-Eintrag 858, 1603
 Media Bar Playliste Media-Eintrag aktuell 858, 1603
 Media Bar Playliste Nachfolger-Eintrag 859, 1631
 Media Bar Playliste nächsten Eintrag wiedergeben 859, 1854
 Media Bar Playliste Zeiger 859, 1675
 Media Bar Playliste Zeiger Nachfolger-Eintrag 859, 1660
 Media Bar Server 859, 1855
 Media Bar Sichtbarkeit des Media Bar Player 859, 1621
 Media Bar Sichtbarkeit des User-Interfaces der Media Bar 859, 1613
 Media Bar Status des Media Bar Player bezüglich Codec 859, 1666
 Media Bar Status des Media Bar Player bezüglich Individualisierung 859, 1666
 Media Bar Status des Media Bar Player bezüglich Lizenz 859, 1666
 Media Bar Status des Media Bar Player bezüglich Playlist 859, 1666
 Media Bar Url des MediaItem Objektes 860, 1692
 Media Bar User-Interface der Media Bar 859, 1613
 Media Bar Wert des mit dem MediaItem Objekt verbundenen Attributes 861, 862, 1793
 Media Bar Wiedergabe des nächsten Eintrages in der Playliste 859, 1854
 Media Bar Wiedergabe einer Media-Datei 859, 1855
 Media Bar Wiedergabe stoppen 859, 1907
 Media Bar Wiedergabestatus Media Bar Player 859, 1675
 Media Bar Zeiger auf aktuellen Media-Eintrag 858, 1603
 Media Bar Zeiger auf die Playliste 859, 1675
 Media Bar Zeiger auf Nachfolger-Eintrag in der Playliste 859, 1660
 Media Bibliothek 1470
 Media Clip 875, 1469
 Media Datei 874, 1468
 MEDIA="print" 420, 982, 1679
 Media-Datei 858
 Media-Datei Beschreibung 907, 913, 920, 925, 936, 944, 969, 1553
 Media-Datei Copyright 908, 914, 921, 926, 936, 944, 970, 1599
 Media-Datei MIME-Typ 909, 915, 922, 928, 937, 946, 972, 1723
 Media-Datei Name des Autor 907, 913, 920, 925, 936, 944, 969, 1570
 Media-Datei Rating 915, 922, 927, 937, 945, 971, 1678
 Media-Datei Titel 909, 915, 922, 927, 937, 946, 972, 1716
 Media-Datei Url 909, 915, 922, 927, 937, 946, 971, 1695
 Media-Element 887, 911, 917, 919, 924, 930, 934, 948, 956, 957, 973, 1120, 1121, 1987, 1988

MediaItem Objekt 858, 860, 1603
 MediaItem Objekt Anzahl verbundene Attribute 860, 861, 1568
 MediaItem Objekt Dauer in Sekunden 860, 1620
 MediaItem Objekt Media-Datei Dauer in Sekunden 860, 1620
 MediaItem Objekt Name 860, 861, 1659
 MediaItem Objekt Name des verbundenen Attributes 860, 861, 1778
 MediaItem Objekt Url 860, 1692
 MediaItem Objekt Wert des verbundenen Attributes 861, 862, 1793
 Medialeiste 396, 437
 Media-Objekt 865
 Media-Objekte 865, 887
 MediaPlayer Active-X-Control 1287
 Media-Typ für Ausgabe eines Objektes 809, 818
 MediaType 874, 1469
 Medien 702
 Medien Wiedergabeplayer 908, 909, 914, 921, 922, 927, 945, 971, 1672, 1674
 Medien zur Animation per Behavior .style.time2 874
 Medien-Fenster 858
 Medien-Fenster (Media Bar) 858
 medium 987, 988, 989, 991
 Mehrere Frameinhalte gleichzeitig ändern 315, 321, 645, 655
 mehrfaches Auftreten 1185
 Mehrfachselektion 278, 477
 mehrsprachige Stichwortliste 748
 Mehrzeiligkeit des Objekthinhaltes 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 666, 731, 736, 739, 744, 750, 751, 753, 775, 1077, 1188, 1191
 Mehrzeiligkeit Inhalt Body 589, 614, 759, 766
 Meldungsfenster erzeugen 388, 392, 398, 1738, 1753, 1859
 Menü 981
 Menü "Extra-Internet Optionen-Sicherheit" 665
 menubar 397, 439
 Menübar 397, 439
 Menüeintrag 981
 Menüleiste 981
 Menüpunkt Datei Drucken 398, 1859
 Menüs 408, 1903
 meta Objekt 748, 749, 750, 1596, 1634, 1659
 METADATA 1195
 Meta-Datei 875, 1469
 Meta-Tag 509
 META-Tag 49
 METHOD 622, 695
 Methode 102
 Methode des Sendens/Empfangens der Daten an/von den Server 627, 1656
 Methode eines JScript-Objektes 155, 211, 1825
 Methode implementiert 223
 Methode Parameterversorgung 223
 Methode und Prototyping 103
 Methoden 44, 72, 111
 Methoden eines Filters 518
 Microsoft JScript Spezialzeichen 113
 Microsoft Active Channel 1162
 Microsoft Active Channel Channel hinzufügen 1162, 1737
 Microsoft Active Channel Unterschrift 1164, 1830
 Microsoft Active Channel-Datei Unterschrift 1164, 1830
 Microsoft Active Desktop 1162



Microsoft Active Desktop Bild hinzufügen 1162, 1737
 Microsoft Active Desktop Webseite hinzufügen 1162, 1737
 Microsoft Javascript-Versionen 30
 Microsoft JScript 40, 1723
 Microsoft JScript Anweisungen 85, 99
 Microsoft JScript Runtime Fehler 115
 Microsoft JScript Standard-Methoden aller Objekte 149
 Microsoft JScript Syntax Fehler 115
 Microsoft JScript-Maschine Erkennung 40
 Microsoft Virtual Machine Verfügbarkeit 854, 1647
 Microsoft Virtuelle Maschine für Java 412, 1834
 Microsoft Visual Basic for Applications 40
 Microsoft Visual Basic Scripting Edition 40
 Microsoft Windows 16-Bit 1672
 Microsoft Windows 32-Bit 1672
 Microsoft Windows CE 1672
 middle 986, 1033, 1037, 1043, 1048, 1053, 1059, 1065, 1071, 1727
 Millisekunden seit dem 1.1.1970 0 Uhr 159, 213, 1913
 MIME 437, 563, 627, 686, 736, 754, 1555, 1621, 1721
 mime_type 438
 Mimetype 46, 753, 1594
 MIME-Typ 563, 736, 754, 1721
 MIME-Typ der Media-Datei 909, 915, 922, 928, 937, 946, 972, 1723
 Mimetype des Plugins 369, 1169, 1170
 Mimetype des Scriptes 1188, 1723
 Mimetype des StyleSheets 996, 1724
 Mime-Typ Dokument 437
 MIME-Typ text/x-scriptlet 1287
 Mimetypen 1167
 MIME-Typen 686, 753, 1555, 1591
 mimeTypees[] 1167
 MIN_VALUE 204, 1657
 minimale Höhe des Objektes 812, 822, 1937
 minimaler numerischer endlicher Wert > 0 204
 Minor-Version 1167
 Minorversion Browserversion 1169, 1566
 missbilligt 976
 mittel 987, 989
 mittlere Maustaste 1101, 1102, 1106
 mittlere Maustaste gedrückt 1104
 modale oder nicht modale Dialoge 378, 416, 1608
 modaler Dialog 405, 1901
 modales Dialogfenster 405, 1901
 modifiers 1156
 MODULATE 908, 914, 921, 927, 945, 971, 1658
 Modulo 75
 MOTIFNAME 908, 914, 921, 927, 945, 971, 1658
 Motion Picture Experts Group 671, 702
 MotionBlur Filter 537
 Motorola 411, 1601
 Mouse-Eventarten 1157
 Mouse-Eventbehandlung 1157
 Mouse-Eventbehandlung beim Internet Explorer ab 4.x 1157
 Mouse-Eventbehandlung beim Netscape ab 4.x 1159
 Mouse-Event-Eigenschaften 1158, 1159
 mouseout 1147
 MouseOut 1147
 mouseover 1147
 MouseOver 1147
 MOV 671, 702
 move 988, 993, 1617
 Move 1103

Mozilla 1167, 1168
 MP3 907
 MPEG 671, 702
 MPG 671
 MS VM 412, 854, 1647, 1834
 MSSmartTagsPreventParsing 749
 multiple Auswahl select objekt 759, 761, 1658, 1723
 multipart/form-data 695
 MULTIPLE 757, 763
 MultipleSelection 278, 477
 Multipurpose Internet Mail Extension 563, 736, 754, 1721
 Multipurpose Internet Mail Extensions 437, 627, 686, 1555, 1621
 Multipurpose Internet Mail Extension-Typ 908, 914, 921, 927, 945, 971, 1656
 mute 889, 1641
 NACHFOLGENDE Kind 241
 nachfolgendes Kind Zeiger 243, 561, 567, 572, 590, 598, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1660
 Nachkommastellen 199, 204, 205, 1911
 nachliegende Seiten 1165
 nachliegendes Dokument 1165
 nächst tieferer Layer 328
 nächste ganze Zahl oberhalb zahl 199, 1745
 nächste ganze Zahl unterhalb zahl 199, 1775
 nächste Seite laut next laden 1165
 nächsten Track abspielen 928, 932, 940, 947, 955, 1847
 nächster übergeordneter Layer 327
 name 638, 671
 Name 874, 1469
 NAME 222, 239, 255, 267, 306, 325, 435, 448, 450, 466, 467, 565, 570, 571, 593, 622, 633, 634, 635, 636, 637, 638, 641, 670, 671, 685, 695, 712, 748, 827, 1103, 1598, 1721, 1731, 1786, 1799, 1868
 Name Browser 410, 1567
 Name Cookie 177, 425, 489, 1598, 1782, 1784, 1789, 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809, 1812, 1814, 1816, 1818, 1820, 1822
 Name der Eigenschaft 1107, 1676
 Name der Variable 62
 Name des Autor der Media-Datei 907, 913, 920, 925, 936, 944, 969, 1570
 Name des Betriebssystems 1170, 1672
 Name des Browsers 1169, 1566
 Name des Kindes 241, 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661
 Name des Objektes 507, 561, 567, 597, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 736, 739, 753, 760, 1077, 1659
 Name des Plugin 368
 Name des Ziel-Fenster bzw. Ziel-Frame 563, 573, 628, 736
 Namensraum 239, 866
 Namensraum HTML 504
 Namensraum laut XMLNS-Attribut 246, 590, 615, 628, 760, 766, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1195, 1683



Namensraum XML 479
 Namensraum XML-Tag 867
 NAMESPACE 506, 507
 namespace Objekt 507
 namespace Objekt Events 1139, 1960
 namespace Objekt Styles 841
 namespaces Collection 1761
 NaN 54, 57, 78, 204, 1660
 natürlicher Logarithmus zur Basis E 199, 1838
 Navigationsleiste 397, 440
 navigator 1286
 navigator Objekt 281, 283, 370, 378, 382, 767, 776,
 1167, 1169, 1170, 1566, 1583, 1592, 1601, 1657, 1660,
 1665, 1675, 1727, 1834, 1909
 navigator.mimeTypes Collection 451, 613, 756, 1168,
 1169, 1170, 1171
 navigator.plugins Collection 311, 1168, 1169, 1170,
 1171
 navigator.plugins.length 1168
 navigator.plugins[] 1168
 navigator.plugins[].description 1168
 navigator.plugins[].filename 1168
 navigator.plugins[].name 1168
 navigator.userProfile Objekt 1171, 1573, 1738, 1748,
 1763, 1778
 Negation 76
 negativ unendlich 204
 NEGATIVE_INFINITY 204, 1660
 Negativ-Unendlich 1660
 ne-resize 988
 NE-resize 993
 Netscape 1104, 1142, 1147, 1155, 1167
 Netscape Browsererweiterungen 1375
 Netscape CSS 334
 Netscape Javascript-Versionen 30
 Netscape Plugins 1375
 Netscape Script signiert 101
 Netscape Sicherheitseinstellungen 368, 1168
 Netscape signiertes Script 101
 Netscape und Plugins 368, 1168
 Netscape unter 6.x 326
 netscape.security.PrivilegeManager.enablePrivilege()
 361, 1145
 Netscape-Privilegmanager 283, 361, 1146
 Netzwerkname 1435, 1688
 Netzwerk-Online-Status 411, 1665
 Netzwerkverbindung 1170, 1665
 Netzwerkverbindung Status 411, 1665
 neue Zeile 56, 58, 113, 114
 neues Dokument erzeugen 432, 1754
 neues Dokument laden 1167, 1739, 1869
 neuladen den Dokumentes 222
 neuladen des Dokumentes 1103
 neuladen des HTML-Dokumentes 1103
 new 752
 new Anweisung 69, 149, 160, 208
 new Anweisung A 72
 new Array() 162
 new Array(elemente_liste) 162
 new Boolean([ausdruck]) 175
 new Date() 176
 new Enumerator() 188
 new Error() 192
 new Function() 73
 new Image() 454, 684, 707
 new Layer() 327

new Number() 199
 new Operator 78, 93
 new RegExp() 1183
 new String("wert_aus_zeichen") 213
 new VBArray() 1428, 1429, 1833, 1835
 newline 56, 58, 114, 1445, 1914
 next 1165
 nicht benutzten Links Farbe 427, 1651
 nicht druckbare Zeichen 114
 nicht dünn und nicht fett 987, 989
 nicht mitscrollende Objekte 676
 nicht numerischer Wert 204
 nicht sicherheitsrelevante Tastenkombinationen 439
 nicht vordefinierte Farbangaben 980
 nichtaktive Fenstertitelzeile 981
 Nichtmedia-Objekt 865
 nicht-modaler Dialog 408, 1903
 nicht-modales Dialogfenster 408, 1903
 NICHTS scannen 50
 No Title 749
 NOBR 815, 825, 1946
 Nodes 222
 noframe Objekt Events 1139, 1960
 noframe Objekt Styles 841
 NOFRAMES 750
 noFrames Objekt 750
 none 770, 1017, 1617, 1683, 1723, 1764
 Normalisierung des DOM 239, 259, 484, 564, 569, 574,
 592, 600, 603, 612, 617, 621, 630, 650, 661, 664, 669,
 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728,
 733, 738, 741, 747, 751, 755, 762, 768, 777, 1025,
 1031, 1035, 1039, 1044, 1050, 1055, 1061, 1067, 1073,
 1081, 1190, 1194, 1196, 1848
 Normaltext-Dokument 438
 NOSCRIPT 751
 noScript Objekt 751
 noscript Objekt Events 1139, 1960
 noscript Objekt Styles 841
 not 75
 Not a Number 57
 Notepad 49
 nowrap 987, 990
 n-resize 988
 N-resize 993
 NS 772
 NS 6.x 604, 772
 NS und Sound 581
 NTFS-Zugriffsbeschränkungen 1446
 null 57, 61, 65, 78, 143, 328, 1147
 null-Zeiger 57, 61, 143
 Null-Zeiger 61
 null-Zuweisung 143
 number 54, 77, 81, 143
 number Basis-Datentyp 57
 number Datentyp 57
 Number JScript-Objekt 149
 Number Objekt 149, 160
 Number Script-Objekt 57, 160, 199, 208, 1654, 1657,
 1660, 1675, 1909, 1910, 1911
 numerische Konvertierung 200
 numerische Liste 985
 Numerischen Wert als Zeichenkette liefern 1828, 1911
 numerischer Datentyp 57
 numerisches Literal 205
 Nummer der Komposition eines Filters 512, 2002
 Nummer Filter 517, 2007



Nummer Run-Time-Error 194, 1662
 Nummernkreuz 561, 572, 1167, 1631
 nur beim Druck anzeigen 420, 982, 1679
 NUR bestimmte Daten auf dem Server dürfen gescannt werden 50
 nw-resize 988
 NW-resize 993
 nzeichenfolge 1184
 oberer Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 oberer Rand des Objektes 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 oberste Fenster 396, 438
 oberstes Fenster in der Fenster-Hierarchie 282
 oberstes Fenster laut Fensterhierarchie 388, 1721
 object 54, 77, 81, 143
 OBJECT 208, 369, 752, 1169, 1170, 1286
 Object JScript-Objekt 62, 149, 208
 object Objekt 752
 object Objekt Events 1139, 1960
 object Objekt Filter 553
 object Objekt Internet Media Typ 753, 1594
 object Objekt Mimetyp 753, 1594
 object Objekt Styles 841
 object Objekt Url des Dokumentes 752, 1579
 Object Script-Objekt 58, 149, 160
 Objekt ... unterstützt Eigenschaft nicht 66, 114
 Objekt nicht gefunden 66, 114
 Objekt .style.time2 1147
 Objekt a 448, 559, 565, 734
 Objekt a Events 1132, 1953
 Objekt a Filter 548
 Objekt a Pseudoklasse des Link, der bereits aktiviert wurde 809, 818, 1917
 Objekt a Pseudoklasse des Link, der nicht kürzlich aktiviert wurde 809, 818, 1917
 Objekt a Pseudoklasse für aktiven Link 809, 818, 1917
 Objekt a Pseudoklasse für Link, der nicht aktiviert wurde 809, 818, 1917
 Objekt a Styles 828
 Objekt Abstand des Aussenrandes zur Umgebung 589, 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Objekt Abstand des Inhaltes zum Aussenrand 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Objekt Abstand linker Objektrand zur Umgebung rechte Kante 811, 813, 814, 817, 821, 823, 824, 1934, 1940
 Objekt Abstand oberer Objektrand zur Umgebung Unterkante 813, 814, 815, 817, 823, 824, 825, 1940, 1941
 Objekt Abstand rechter Objektrand zur Umgebung linke Kante 813, 814, 817, 823, 824, 1940, 1941, 1942
 Objekt Abstand Textzeichen 811, 821, 1934
 Objekt Abstand unterer Objektrand zur Umgebung Oberkante 813, 817, 823, 1940
 Objekt Abstand zum linken Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759,

765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Objekt Abstand zum oberen Rand des Fensters 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Objekt Abstand zweier Objekte oder zweier Textzeilen 811, 821, 1934
 Objekt Abstand zwischen Objekt und Margin bzw. Rahmen links 813, 823, 1938
 Objekt Abstand zwischen Objekt und Margin bzw. Rahmen links, rechts, oben, unten 812, 822, 1938
 Objekt Abstand zwischen Objekt und Margin bzw. Rahmen oben 813, 823, 1939
 Objekt Abstand zwischen Objekt und Margin bzw. Rahmen rechts 813, 823, 1939
 Objekt Abstand zwischen Objekt und Margin bzw. Rahmen unten 812, 1938
 Objekt ActiveXObject 1420
 Objekt Aktion in der Timeline 610, 885, 909, 915, 922, 927, 932, 946, 954, 972, 1711
 Objekt aktivieren 560, 571, 597, 609, 614, 618, 627, 666, 681, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 743, 765, 774, 885, 893, 897, 904, 907, 913, 920, 926, 936, 944, 949, 954, 965, 970, 1014, 1027, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1191, 1579
 Objekt Aktivierung 1112, 1977
 Objekt an Pixelpos 434, 1764
 Objekt anzeigen transparent 811, 821, 1935
 Objekt animatecolor 893, 897, 898, 904, 965, 1562, 1729
 Objekt animatemotion 904, 1666, 1670
 Objekt animieren 508, 907
 Objekt anklicken 1098
 Objekt Anzeige Art 810, 816, 820, 825, 1930, 1948
 Objekt Anzeige mit Scrollelementen 812, 822, 1937
 Objekt anzeigen 382, 1662
 Objekt anzeigen horizontal 810, 820, 1930
 Objekt applet 566
 Objekt applet Events 1132, 1953
 Objekt applet Filter 548
 Objekt applet Styles 829
 Objekt area 455, 571, 739, 742
 Objekt area Events 1132, 1953
 Objekt area Filter 548
 Objekt area Styles 829
 Objekt arguments 149, 160
 Objekt Array 149, 160
 Objekt Array aus Literalen 74
 Objekt Art Anzeige 810, 816, 820, 825, 1930, 1948
 Objekt Art der Objektpositionierung innerhalb Eltern 814, 817, 824, 1941
 Objekt asiatische Zeichen Textfluss 811, 820, 1933
 Objekt attribute 255, 266, 463, 464, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 751, 755, 761, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1196, 1778
 Objekt Attribute 246, 1693
 Objekt Attribute-Referenzen 464
 Objekt Attributwert 247, 599, 686, 688, 697, 701, 705, 709, 714, 718, 723, 727, 761, 767, 1078, 1727
 Objekt auf Objektklasse prüfen 78
 Objekt auf Printmedium ausgeben 1655



Objekt aus einem Objekt entfernen 260, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 741, 747, 762, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1194, 1867

Objekt Ausgabe auf Bildschirm 1655

Objekt Ausgabe auf Printmedium 1655

Objekt Ausgabe eines Objektes 809, 818

Objekt auslösen eines Events 1772

Objekt Ausrichtung 1014, 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069, 1562

Objekt Ausrichtung des Inhaltes 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1727

Objekt Ausrichtung vertikal 816, 825, 1947

Objekt Ausschnitt Koordinate links 849, 1593

Objekt Ausschnitt Koordinate oben 849, 1594

Objekt Ausschnitt Koordinate rechts 849, 1594

Objekt Ausschnitt Koordinate unten 849, 1593

Objekt Aussenrand Abstand links zum angrenzenden Objekt 812, 821, 1936

Objekt Aussenrand Abstand oben zum angrenzenden Objekt 812, 821, 1936

Objekt Aussenrand Abstand rechts zum angrenzenden Objekt 812, 821, 1936

Objekt Aussenrand Abstand unten zum angrenzenden Objekt 811, 821, 1936

Objekt Aussenrand links, rechts, oben, unten 811, 821, 1935

Objekt Aussenrand und Abstand des Inhaltes 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939

Objekt Baumhierarchie 222

Objekt Behavior 507, 1761

Objekt Behavior dynamisch verwalten 1761

Objekt betriebssystem-nah 225

Objekt bewegen 1122, 1989

Objekt Bezeichner 481, 561, 567, 572, 582, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635

Objekt bgsound 575, 885, 890, 909, 915, 922, 928, 932, 946, 955, 972, 1658, 1732

Objekt bgsound 873

Objekt bgsound Events 1132, 1954

Objekt bgsound mehrere im Dokument 576

Objekt bgsound Styles 829

Objekt Bildschirmausgabe 1655

Objekt body 456, 457, 458, 504, 770, 771, 772, 864, 866, 885, 887, 890, 894, 895, 898, 900, 905, 906, 908, 909, 911, 914, 915, 917, 921, 922, 924, 927, 928, 930, 932, 934, 945, 946, 948, 949, 950, 951, 954, 955, 957, 966, 967, 971, 972, 974, 1082, 1125, 1128, 1658, 1709, 1732, 1883, 1992, 1995

Objekt BODY 424, 1560

Objekt body Events 1132, 1954

Objekt body Filter 548

Objekt body Styles 829

Objekt Boolean 149, 160

Objekt Breite 682, 745, 754, 813, 814, 816, 817, 823, 824, 826, 1940, 1941, 1948

Objekt browserintern 37, 68

Objekt button 457, 458, 593, 622, 632, 685, 686, 770, 771, 1082, 1883

Objekt button Events 1133, 1954

Objekt button Filter 549

Objekt button Styles 830

Objekt clipboardData 378, 1108, 1593

Objekt Collection document.all 610, 775

Objekt command 275, 474

Objekt comment 53, 601, 1650

Objekt Comment 252, 432, 1754

Objekt comment Events 1133, 1954

Objekt comment Styles 830

Objekt currentStyle 797, 847, 849, 1593, 1594

Objekt currentStyle Styles 830

Objekt currentStyle Filter 549

Objekt currTimeState 560, 561, 562, 563, 571, 572, 573, 597, 598, 599, 604, 609, 610, 614, 615, 618, 619, 627, 628, 666, 667, 681, 682, 686, 687, 688, 691, 692, 695, 696, 697, 700, 703, 704, 708, 709, 712, 713, 714, 717, 718, 721, 722, 725, 726, 727, 743, 744, 745, 765, 767, 773, 774, 775, 776, 887, 890, 943, 1014, 1018, 1028, 1029, 1041, 1042, 1046, 1048, 1051, 1053, 1057, 1059, 1063, 1065, 1069, 1071, 1076, 1077, 1078, 1107, 1114, 1117, 1120, 1121, 1124, 1125, 1127, 1128, 1129, 1131, 1191, 1192, 1553, 1555, 1557, 1559, 1560, 1562, 1568, 1573, 1579, 1583, 1584, 1586, 1588, 1590, 1593, 1603, 1606, 1616, 1617, 1619, 1621, 1623, 1626, 1628, 1629, 1630, 1631, 1632, 1633, 1636, 1641, 1642, 1645, 1646, 1647, 1648, 1649, 1654, 1655, 1656, 1657, 1658, 1666, 1669, 1670, 1671, 1673, 1675, 1676, 1680, 1681, 1686, 1690, 1691, 1693, 1696, 1698, 1703, 1705, 1706, 1707, 1709, 1711, 1713, 1718, 1719, 1723, 1725, 1726, 1728, 1729, 1730, 1732, 1734, 1736, 1742, 1743, 1760, 1764, 1765, 1845, 1847, 1850, 1851, 1853, 1855, 1857, 1871, 1873, 1877, 1878, 1879, 1880, 1882, 1904, 1979, 1984, 1987, 1988, 1991, 1992, 1994, 1995, 1996, 1997, 1999, 2000

Objekt custom 479, 801

Objekt custom Events 1133, 1954

Objekt custom Filter 549

Objekt custom Styles 830

Objekt das das Event auslöst 1107, 1695

Objekt Date 149, 160

Objekt Datei und Pfad 562, 572, 1167, 1671

Objekt Daten 601, 1603

Objekt Daten nicht permanent speichern 863, 864

Objekt Daten speichern permanent 862

Objekt Daten speichern über mehrere Onlinesitzungen 862

Objekt Deaktivierung 1114, 1116, 1979, 1982

Objekt des HTML-Tag 418

Objekt Dictionary 1427

Objekt Dimension 812, 822, 1937

Objekt div 603

Objekt div Events 1133, 1954

Objekt div Filter 549

Objekt div Styles 831

Objekt document 281, 283, 370, 381, 428, 801, 1614, 1714, 1754

Objekt document des Netscape 304

Objekt document Events 1133, 1955

Objekt document Filter 549

Objekt document Styles 832

Objekt document.body 584, 866

Objekt document.form 622

Objekt document.form.input 632, 685

Objekt document.form.input.button 632

Objekt document.form.input.checkbox 633

Objekt document.form.input.fileupload 634



Objekt document.form.input.hidden 634
 Objekt document.form.input.password 635
 Objekt document.form.input.radio 635
 Objekt document.form.input.reset 637
 Objekt document.form.input.submit 638
 Objekt document.form.input.text 638
 Objekt document.html 662
 Objekt document.namespace 504, 851, 852, 855, 856, 857, 862, 863, 864
 Objekt document.select 757, 759, 760, 761, 763, 765, 766, 767, 1607, 1635, 1649, 1658, 1666, 1687, 1692, 1711, 1723
 Objekt document.select.ListBox 766, 1635
 Objekt document.select.option 763, 1666
 Objekt document.select.option Default-Selektionsstatus 765, 1607
 Objekt document.select.option interner Wert 767, 1711
 Objekt document.select.option Label 766, 1649
 Objekt document.select.option laufende Nummer in List-Box 765, 1635
 Objekt document.select.option Selektionsstatus 766, 1687
 Objekt document.selection 456, 457, 458, 770, 771, 1083, 1723, 1746, 1757, 1764, 1883
 Objekt document.selection löschen Selektion 770, 1764
 Objekt document.selection Markierung der Selektion 770, 1746
 Objekt document.selection Selektion löschen 770, 1764
 Objekt document.selection Selektion Markierung 770, 1746
 Objekt document.selection Selektion Typ 770, 1723
 Objekt document.selection Selektion Universal-Bereich erzeugen 458, 770, 1083, 1757
 Objekt document.selection Typ der Selektion 770, 1723
 Objekt document.selection Universal-Bereich erzeugen Selektion 458, 770, 1083, 1757
 Objekt dokument Kompatibilität des IE 6.x zu CSS1 424, 1595
 Objekt Drag und Drop 1620
 Objekt Druckvorschau 1655
 Objekt durch anderes Objekt ersetzen 240
 Objekt durch anderes Objekt komplett ersetzen 262, 593, 601, 603, 612, 617, 621, 630, 662, 665, 669, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 769, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1194, 1870
 Objekt Eigenschaft ändern 1126, 1993
 Objekt Eigenschaften und Methoden 111
 Objekt element 490, 499, 501, 502, 503, 504, 851, 1147, 1755, 1770, 1772, 1981, 1983, 1993
 Objekt embed Events 1134, 1955
 Objekt embed Styles 832
 Objekt Enumerator 149, 160
 Objekt Error 149, 160
 Objekt ersetzen durch ein Objekt 262, 593, 601, 603, 612, 617, 621, 630, 662, 664, 669, 690, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 742, 747, 763, 769, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1068, 1074, 1081, 1194, 1870
 Objekt erste Textzeile 809, 818, 1917
 Objekt erster Buchstabe 809, 818, 1917
 Objekt Erzeugung durch Makro 275, 474
 Objekt event 41, 281, 283, 370, 381, 1087, 1095, 1623, 1772
 Objekt Event auslösen 1772
 Objekt event Styles 832

Objekt event und Maus 1107, 1630
 Objekt event und Tastatur 1107, 1647
 Objekt event Url laut einem Kommando aus einer Advanced Streaming Format (ASF) Datei 917, 930, 1107, 1131, 1726, 2000
 Objekt event.dataTransfer 378, 413, 1108, 1593, 1617, 1620
 Objekt Eventarten 1131
 Objekt externes 70
 Objekt Farbe des 3D-Rahmens 1014, 1058, 1063, 1069, 1581
 Objekt Farbe eines Objektes 618, 1594
 Objekt fieldSet 614
 Objekt fieldset Events 1134, 1955
 Objekt fieldset Filter 549
 Objekt fieldset Styles 832
 Objekt FileSystemObject 1430
 Objekt FileSystemObject.Drive 1434
 Objekt FileSystemObject.File 1433, 1437, 1788
 Objekt FileSystemObject.Folder 1433, 1439, 1788, 1802
 Objekt FileSystemObject.TextStream 1442
 Objekt filter 508, 810, 820, 878, 1931
 Objekt filter Feld aller Filter 511
 Objekt Filter-Animation 508
 Objekt Flussrichtung bei Objektanzeige 816, 826, 1949
 Objekt Focussierbarkeit 481, 561, 567, 572, 609, 618, 627, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 744, 753, 775, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1191, 1632
 Objekt font 618
 Objekt font Events 1134, 1955
 Objekt font Filter 550
 Objekt font Größe 619, 1691
 Objekt font Styles 833
 Objekt font Typ des Fonts 618, 1625
 Objekt form 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 1607
 Objekt Form 562, 573, 1688
 Objekt form Events 1134, 1956
 Objekt form Filter 550
 Objekt form Styles 833
 Objekt form.input image 271, 471
 Objekt frame 453, 642, 651
 Objekt frame Filter 550
 Objekt frame Styles 833
 Objekt frameset 652
 Objekt frameset Events 1135, 1956
 Objekt frameset Filter 550
 Objekt frameset Styles 834
 Objekt füllen mit Daten 428, 482, 507, 562, 568, 572, 598, 602, 619, 628, 649, 660, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 775, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1679
 Objekt Function 95, 149, 160
 Objekt Gestalt 562, 573, 1688
 Objekt head Events 1135, 1956
 Objekt head Styles 834
 Objekt Hintergrund 809, 818, 1918
 Objekt Hintergrundbild 1014, 1027, 1041, 1063, 1069, 1574
 Objekt history 281, 283, 370, 382, 1165, 1167, 1633, 1739, 1869
 Objekt Höhe 648, 666, 681, 744, 753, 811, 813, 817, 820, 823, 1016, 1058, 1064, 1070, 1632, 1932, 1940



Objekt Höhe minimale 812, 822, 1937
 Objekt horizontal anzeigen 810, 820, 1930
 Objekt horizontal rendern 810, 820, 1930
 Objekt html 418
 Objekt html comment 665
 Objekt html comment Events 1135, 1956
 Objekt html comment Styles 835
 Objekt html Events 1135, 1956
 Objekt html Styles 835
 Objekt HTML-Attribute 250, 483, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 697, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 754, 761, 767, 776, 826, 1019, 1030, 1034, 1038, 1043, 1048, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1746
 Objekt ID 247, 430, 481, 561, 563, 567, 568, 572, 573, 590, 597, 599, 601, 602, 609, 615, 618, 619, 627, 628, 648, 649, 659, 660, 662, 663, 666, 667, 681, 682, 687, 688, 691, 692, 696, 697, 700, 701, 703, 704, 708, 709, 713, 714, 717, 718, 721, 722, 726, 727, 730, 731, 736, 737, 739, 740, 744, 745, 750, 751, 753, 754, 761, 767, 775, 776, 1016, 1018, 1028, 1029, 1032, 1033, 1036, 1037, 1041, 1042, 1046, 1048, 1052, 1053, 1058, 1059, 1064, 1065, 1070, 1071, 1077, 1078, 1188, 1191, 1192, 1195, 1635, 1724
 Objekt ID_Player 1478
 Objekt ID_Player.closedCaption 1484
 Objekt ID_Player.controls 1484
 Objekt ID_Player.currentMedia 1486
 Objekt ID_Player.currentPlaylist 1488
 Objekt ID_Player.error 1491
 Objekt ID_Player.network 1493
 Objekt ID_Player.settings 1496
 Objekt iframe 665, 816, 826, 1950
 Objekt iframe Events 1135, 1956
 Objekt iframe Filter 550
 Objekt iframe Styles 835
 Objekt im Dokument 255, 435, 827, 1786
 Objekt im Offscreen rendern 382, 1662
 Objekt img 622, 670, 685, 703, 738, 812, 822, 823, 1938
 Objekt img Events 1135, 1956
 Objekt img Filter 551
 Objekt img Styles 835
 Objekt implementation 427, 1635
 Objekt Import Behavior 507, 1761
 Objekt Import Verhalten 507, 1761
 Objekt in eine Objekt einfügen 258, 564, 569, 574, 592, 600, 602, 611, 617, 621, 630, 650, 661, 664, 669, 684, 689, 694, 698, 701, 706, 710, 715, 719, 724, 728, 733, 738, 741, 746, 755, 762, 768, 777, 1021, 1030, 1035, 1038, 1044, 1049, 1054, 1061, 1067, 1073, 1081, 1189, 1194, 1829
 Objekt in Javascript vordefiniert 160
 Objekt Index in der Collection document.all 482, 598, 602, 628, 649, 660, 663, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 736, 740, 745, 750, 754, 1000, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1188, 1692
 Objekt Inhalt Ausrichtung 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1727
 Objekt Initialisierung mit Standardwert 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 1607
 Objekt input 248, 593, 622, 632, 685, 1607, 1727
 Objekt input button 686, 1722
 Objekt input button Events 1135, 1957
 Objekt input button Filter 551

Objekt input button Styles 835
 Objekt input checkbox 248, 690, 1078, 1607, 1699, 1722, 1727
 Objekt input checkbox Events 1136, 1957
 Objekt input checkbox Filter 551
 Objekt input checkbox Grauzustand (Dimmed) und Selektiertheit 691, 1635
 Objekt input checkbox Selektiertheit 691, 1635
 Objekt input checkbox Selektionsstatus 691, 712, 713, 1591, 1606
 Objekt input checkbox Style 690
 Objekt input checkbox Styles 836
 Objekt input Events 1135, 1957
 Objekt input file 248, 695, 1607, 1722, 1727
 Objekt input file Events 1136, 1957
 Objekt input file Filter 551
 Objekt input file Styles 836
 Objekt input Filter 551
 Objekt input Focus 690, 694, 699, 707, 711, 716, 720, 725, 729, 1883
 Objekt input hidden 248, 699, 1607, 1722, 1727
 Objekt input hidden Events 1136, 1957
 Objekt input hidden Filter 552
 Objekt input hidden Styles 837
 Objekt input im Formular markieren 690, 694, 699, 706, 711, 716, 720, 725, 729, 1883
 Objekt input image 671, 702, 722, 1722
 Objekt input image Events 1136, 1957
 Objekt input image Filter 552
 Objekt input image Styles 837
 Objekt input markieren 690, 694, 699, 706, 711, 716, 720, 725, 729, 1883
 Objekt input password 248, 707, 1607, 1722, 1727
 Objekt input password Events 1136, 1958
 Objekt input password Filter 552
 Objekt input password Styles 837
 Objekt input radio 248, 712, 1607, 1722, 1727
 Objekt input radio Events 1137, 1958
 Objekt input radio Filter 552
 Objekt input radio Selektionsstatus 691, 712, 713, 1591, 1606
 Objekt input radio Styles 838
 Objekt input reset 248, 716, 1607, 1722, 1727
 Objekt input reset Events 1137, 1958
 Objekt input reset Filter 552
 Objekt input reset Styles 838
 Objekt input Styles 835
 Objekt input submit 248, 721, 1607, 1722, 1727
 Objekt input submit Events 1137, 1958
 Objekt input submit Filter 553
 Objekt input submit Styles 839
 Objekt input text 248, 457, 458, 725, 770, 771, 1082, 1607, 1722, 1727, 1883
 Objekt input text Events 1137, 1959
 Objekt input text Filter 553
 Objekt input text Styles 839
 Objekt input Typ 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 Objekt input Varinate 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 Objekt instanziiert 69
 Objekt Interaktionsfähigkeit 481, 482, 561, 567, 572, 582, 589, 597, 601, 609, 614, 618, 627, 648, 659, 662, 666, 681, 686, 687, 691, 696, 700, 703, 708, 713, 717, 721, 725, 726, 730, 736, 739, 743, 744, 749, 750, 751, 753, 759, 765, 766, 774, 775, 1016, 1017, 1028, 1032,



1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1610, 1640
 Objekt internes ID 481, 561, 567, 572, 597, 601, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 Objekt Kind 260, 592, 600, 603, 612, 617, 621, 630, 661, 664, 669, 689, 694, 699, 702, 706, 711, 716, 720, 724, 729, 733, 741, 747, 762, 768, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1194, 1867
 Objekt Kinder des Objektes 268, 271, 467, 471
 Objekt klonen 239, 251, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1749
 Objekt Kontextänderung 1980
 Objekt Koordinate des Ausschnittes links 849, 1593
 Objekt Koordinate des Ausschnittes oben 849, 1594
 Objekt Koordinate des Ausschnittes rechts 849, 1594
 Objekt Koordinate des Ausschnittes unten 849, 1593
 Objekt label 248, 729, 1607, 1722, 1727
 Objekt label Events 1138, 1959
 Objekt label Styles 839
 Objekt laden 369, 567, 752, 1168, 1565
 Objekt Ladezustand 681, 686, 703, 1595
 Objekt layer 604, 816, 826, 1950
 Objekt Layer 772
 Objekt Layout-Komponente 483, 563, 568, 573, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 750, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751
 Objekt link 559, 734
 Objekt link Events 1138, 1959
 Objekt link Styles 840
 Objekt linke obere Ecke 459, 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1083, 1191, 1664
 Objekt linke obere Ecke 459, 482, 561, 567, 572, 598, 610, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1083, 1191, 1663
 Objekt linker Rand 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 714, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 Objekt Listendarstellung Aufzählungsliste-Elemente 811, 821, 1934
 Objekt Listendarstellung Marker als Bild 811, 821, 1935
 Objekt Listendarstellung Marker als Image 811, 821, 1935
 Objekt Listendarstellung Marker Position 811, 821, 1935
 Objekt Listendarstellung Markertyp (nicht Image) 811, 821, 1935
 Objekt location 281, 283, 370, 382, 419, 1166, 1651
 Objekt map 455, 571, 738, 742
 Objekt map Events 1138, 1959

Objekt map Styles 840
 Objekt marquee 742
 Objekt marquee Anzahl der Scrollaktionen 744
 Objekt marquee endlos scrollen 1653
 Objekt marquee Events 1138, 1959
 Objekt marquee Filter 553
 Objekt marquee Scrollaktionen Anzahl 744
 Objekt marquee Scrollaktionen Stop 748, 1907
 Objekt marquee Scrollgeschwindigkeit 744, 1684
 Objekt marquee Scrollrichtung 1610
 Objekt marquee Scrollrichtung beim Start 743, 1610
 Objekt marquee Scrollschrittweite 744, 1684
 Objekt marquee Scroll-Takteinheit 745, 1721
 Objekt marquee Scrollverhalten 743
 Objekt marquee Start Scrollaktion 747, 1906
 Objekt marquee Styles 840
 Objekt Math 149, 160
 Objekt Mauscursor Art 810, 820, 1930
 Objekt Mausüberwachung 1120, 1987
 Objekt Maus-Überwachung ausschalten 446, 484, 564, 569, 575, 592, 600, 612, 617, 621, 630, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 741, 747, 755, 762, 768, 777, 1026, 1031, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1107, 1194, 1865
 Objekt Maus-Überwachung einschalten 484, 565, 570, 575, 593, 601, 612, 618, 622, 631, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 742, 747, 756, 763, 769, 778, 1027, 1031, 1045, 1051, 1056, 1062, 1068, 1074, 1082, 1107, 1194, 1887
 Objekt MediaItem 858, 860, 1603
 Objekt MediaItem Anzahl verbundene Attribute 860, 861, 1568
 Objekt MediaItem Dauer in Sekunden 860, 1620
 Objekt MediaItem Media-Datei Dauer in Sekunden 860, 1620
 Objekt MediaItem Name 860, 861, 1659
 Objekt MediaItem Name des verbundenen Attributes 860, 861, 1778
 Objekt MediaItem Url 860, 1692
 Objekt MediaItem Wert des verbundenen Attributes 861, 862, 1793
 Objekt Media-Typ für Ausgabe eines Objektes 809, 818
 Objekt meta 748, 749, 750, 1596, 1634, 1659
 Objekt minimale Höhe 812, 822, 1937
 Objekt mit Focus 424, 1560
 Objekt mit HREF-Eigenschaft 455, 738
 Objekt mit HTML-Tags 242, 481, 560, 567, 571, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 730, 736, 739, 743, 750, 751, 752, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1069, 1076, 1187, 1191, 1195, 1590
 Objekt mit Style-Layout 849, 1631
 Objekt Name 507, 561, 567, 597, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 736, 739, 753, 760, 1077, 1659
 Objekt namespace 507
 Objekt namespace Events 1139, 1960
 Objekt namespace Styles 841
 Objekt navigator 281, 283, 370, 378, 382, 767, 776, 1167, 1169, 1170, 1566, 1583, 1592, 1601, 1657, 1660, 1665, 1675, 1727, 1834, 1909
 Objekt navigator.userProfile 1171, 1573, 1738, 1748, 1763, 1778
 Objekt neu laden 369, 1168
 Objekt nicht permanentes Speichern von Daten 863, 864



Objekt noframe Events 1139, 1960
 Objekt noframe Styles 841
 Objekt noFrames 750
 Objekt noScript 751
 Objekt noscript Events 1139, 1960
 Objekt noscript Styles 841
 Objekt Number 149, 160
 Objekt oberer Rand 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 Objekt object 752
 Objekt object Events 1139, 1960
 Objekt object Filter 553
 Objekt object Internet Media Typ 753, 1594
 Objekt object Mimetyp 753, 1594
 Objekt object Styles 841
 Objekt object Url des Dokumentes 752, 1579
 Objekt objektübergreifende Operationen 142
 Objekt objektunabhängige Operationen 142
 Objekt Operationen objektübergreifend 142
 Objekt Operationen objektunabhängig 142
 Objekt Operationen vordefiniert 142
 objekt option 248, 463, 465, 1727
 Objekt option Events 1139, 1960
 Objekt option Filter 553
 Objekt option Styles 842
 Objekt page 818, 996, 998, 1553, 1678, 1687, 1738, 1917
 Objekt permanent Daten speichern 862
 Objekt Pixelposition des Rechteckes um Objekt 462, 1087, 1581, 1649, 1682, 1720
 Objekt Plain-Text 482, 561, 568, 572, 598, 601, 610, 615, 619, 628, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1667
 Objekt playItem 911
 Objekt playItem aktiv 939, 1561
 Objekt playItem aktivieren 928, 929, 932, 933, 940, 947, 955, 1847, 1857
 Objekt playItem Index in der Collection playList 1635
 Objekt PlaylistInfo 859, 860, 861, 1631, 1660
 Objekt popup Events 1139, 1960
 Objekt popup Styles 842
 Objekt Position sichtbarer Bereich über dem Objekt 810, 820, 1928
 Objekt priorityClass 918
 Objekt privat 68
 Objekt Prototyp-Bereich 150, 209, 1677
 Objekt Prototyping 150, 160, 209, 1677
 Objekt Pseudoklasse für Style der ersten Zeile 809, 818, 1917
 Objekt Pseudoklasse für Style des ersten Buchstaben 809, 818, 1917
 Objekt Rahmen Art alle 4 Seiten 810, 819, 1925
 Objekt Rahmen Art links 809, 819, 1923
 Objekt Rahmen Art oben 810, 819, 1926
 Objekt Rahmen Art rechts 810, 819, 1924
 Objekt Rahmen Art und Dicke alle 4 Seiten 809, 819, 1920
 Objekt Rahmen Art und Dicke links 809, 819, 1922
 Objekt Rahmen Art und Dicke oben 810, 819, 1925
 Objekt Rahmen Art und Dicke rechts 810, 819, 1924
 Objekt Rahmen Art und Dicke unten 809, 819, 1921

Objekt Rahmen Art unten 809, 819, 1921
 Objekt Rahmen Dicke alle 4 Seiten 810, 819, 1927
 Objekt Rahmen Dicke links 809, 819, 1923
 Objekt Rahmen Dicke oben 810, 819, 1926
 Objekt Rahmen Dicke rechts 810, 819, 1925
 Objekt Rahmen Dicke unten 809, 819, 1922
 Objekt Rahmen Farbe alle 4 Seiten 809, 819, 1922
 Objekt Rahmen Farbe links 809, 819, 1923
 Objekt Rahmen Farbe oben 810, 819, 1926
 Objekt Rahmen Farbe rechts 810, 819, 1924
 Objekt Rahmen Farbe unten 809, 819, 1921
 Objekt Rahmen Singleborder 809, 819, 1922
 Objekt Rahmenfarbe 648, 659, 1014, 1057, 1063, 1069, 1581
 Objekt rechte untere Ecke 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1037, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1192, 1664
 Objekt rechte unteren Ecke 482, 561, 567, 572, 598, 609, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1191, 1662
 Objekt Referenz als absoluter Pfad 48
 Objekt Referenz auf TextRectangle 460, 483, 564, 569, 574, 600, 602, 611, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1778
 Objekt Referenz aus Teilzeigern 48
 Objekt Referenzierung 48, 582
 Objekt regexp 114, 1183
 Objekt RegExp 114, 149, 160, 218, 1869
 Objekt Reihenfolge überlappender Objekte 816, 826, 1950
 Objekt rendern 223
 Objekt rendern horizontal 810, 820, 1930
 Objekt runtimeStyle 849
 Objekt runtimeStyle Filter 553
 Objekt runtimeStyle Styles 842
 Objekt screen 281, 283, 370, 383, 1182, 1183, 1574, 1583, 1594, 1607, 1608, 1632, 1652, 1684, 1725, 1733
 Objekt screen Styles 842
 Objekt script 1187, 1188, 1190, 1607, 1623, 1634
 Objekt script Events 1139, 1960
 Objekt script Styles 842
 Objekt Script-Function 106
 Objekt Scrollbalken Bild Farbe 824
 Objekt Scrollbalken Elemente Farbe 814, 824, 1942
 Objekt Scrollbalken Farbe 824
 Objekt Scrollbalken Kante Farbe 824
 Objekt Scrollbalken Kanten Farbe 824
 Objekt Scrollbalken Pfeil Farbe 824
 Objekt Scrollbalken Pfeil und Bild Farbe 824
 Objekt Scrollbalken Pfeil und Kanten Farbe 824
 Objekt Scrollbalken Rinne Farbe 824
 Objekt Scrollbalken Ziehelement Farbe 815, 824, 1944
 Objekt Scrollelemente 812, 822, 1937
 Objekt scrollen 457, 460, 484, 564, 570, 575, 601, 603, 612, 617, 621, 631, 669, 684, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 771, 778, 812, 822, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1085, 1194, 1876, 1937
 Objekt Seitenumbruch nach dem Druck des Objektes 813, 823, 1939



Objekt Seitenumbruch vor dem Druck des Objektes 813, 823, 1939
 objekt select 248, 463, 465, 1727
 Objekt select 622
 objekt select Auswahl multible 759, 761, 1658, 1723
 Objekt select Events 1139, 1960
 Objekt select Filter 554
 Objekt select Index der Option 760, 1687
 Objekt select List-Box 760, 1692
 objekt select multible Auswahl 759, 761, 1658, 1723
 Objekt select Optionindex 760, 1687
 Objekt selection 457, 458, 770, 771, 1082, 1883
 Objekt selection Events 1139, 1960
 Objekt selection Filter 554
 Objekt Selektierbarkeit 481, 482, 560, 563, 568, 571, 573, 597, 599, 609, 610, 618, 619, 627, 628, 648, 649, 659, 660, 666, 667, 680, 682, 686, 688, 691, 692, 695, 697, 700, 701, 703, 705, 708, 709, 712, 714, 717, 718, 721, 723, 725, 727, 730, 731, 743, 745, 754, 774, 1014, 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1065, 1069, 1076, 1078, 1191, 1192, 1568, 1725
 Objekt selektieren 460, 1082, 1085, 1883
 Objekt sichtbar 457, 460, 484, 564, 570, 575, 601, 603, 612, 617, 621, 631, 669, 684, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 771, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1085, 1194, 1876
 Objekt Sichtbarkeit 810, 816, 820, 825, 1930, 1948
 Objekt Sichtbarkeit mit Platz im Layout der Umgebung 810, 820, 1930
 Objekt Sichtbarkeit ohne Platz im Layout der Umgebung 816, 825, 1948
 Objekt span 603, 771
 Objekt span Events 1139, 1960
 Objekt span Filter 554
 Objekt span rendern Internet Explorer 772
 Objekt span rendern Netscape 772
 Objekt span Styles 842
 Objekt Speicherfreigabe 143
 Objekt Sprachcode 561, 736, 1634
 Objekt Sprache 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708
 Objekt Standard-Eigenschaften in JScript 149
 Objekt Standardmethoden 143
 Objekt Standard-Methoden in JScript 149
 Objekt Standardwert 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 1607
 Objekt Status 246, 428, 482, 507, 562, 568, 572, 583, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1126, 1188, 1192, 1195, 1679, 1993
 Objekt String 149, 160, 815, 825, 1946
 Objekt style 424, 479, 735, 778, 826, 847, 849, 994, 1595, 1631, 1723
 Objekt Style Cursorform 337, 781, 828, 1912, 1930, 1952
 Objekt Style Cursorform als Datei 337, 781, 828, 1912, 1930, 1952
 Objekt style Events 1140, 1961
 Objekt style Filter 554
 Objekt Style Filter-Attribut 510

Objekt style Styles 842
 Objekt Style Url eines Behaviors 337, 781
 Objekt Style Wertkonvertierung einer Url 337, 781, 828, 1912, 1951
 Objekt STYLE-Attribut-Wert 810, 817, 820, 1929
 Objekt styleSheet 994, 998, 999, 1553, 1634, 1667, 1668, 1679, 1718, 1724, 1738, 1868
 Objekt styleSheet erzeugen 433
 Objekt styleSheet Styles 842
 Objekt Tabelle Layout 815, 824, 1944
 Objekt table 864, 1000, 1019, 1022, 1023, 1026, 1043, 1044, 1049, 1050, 1054, 1055, 1590, 1591, 1595, 1604, 1627, 1683, 1704, 1754, 1758, 1759, 1773, 1830, 1835, 1841, 1845, 1856, 1865
 Objekt table Events 1140, 1961
 Objekt table Filter 554
 Objekt table Styles 843
 Objekt table.caption 1014, 1027, 1590, 1727
 Objekt table.caption Events 1140, 1961
 Objekt table.caption Filter 554
 Objekt table.caption Styles 843
 Objekt table.col 1032, 1692
 Objekt table.col Events 1140, 1961
 Objekt table.col Filter 554
 Objekt table.col Styles 843
 Objekt table.colGroup 1035, 1692
 Objekt table.colGroup Events 1140, 1961
 Objekt table.colGroup Filter 555
 Objekt table.colGroup Styles 844
 Objekt table.tBody 1019, 1022, 1023, 1040, 1043, 1044, 1049, 1050, 1054, 1055, 1759, 1830, 1841
 Objekt table.tBody Events 1140, 1961
 Objekt table.tBody Filter 555
 Objekt table.tBody Styles 844
 Objekt table.tFoot 1018, 1019, 1022, 1023, 1043, 1044, 1046, 1049, 1050, 1054, 1055, 1711, 1759, 1830, 1841
 Objekt table.tFoot Events 1140, 1962
 Objekt table.tFoot Filter 555
 Objekt table.tFoot Styles 844
 Objekt table.tHead 1018, 1019, 1022, 1023, 1043, 1044, 1049, 1050, 1051, 1054, 1055, 1711, 1759, 1830, 1841
 Objekt table.tHead Events 1141, 1962
 Objekt table.tHead Filter 555
 Objekt table.tHead Styles 845
 Objekt table.tr 1056, 1682, 1685, 1758, 1830
 Objekt table.tr Events 1141, 1962
 Objekt table.tr Filter 555
 Objekt table.tr Styles 845
 Objekt table.tr.td 1062, 1063, 1065, 1069, 1070, 1071, 1590, 1595, 1683
 Objekt table.tr.td Events 1141, 1962
 Objekt table.tr.td Filter 555
 Objekt table.tr.td Styles 845
 Objekt table.tr.th 1063, 1065, 1068, 1070, 1071, 1590, 1595, 1683
 Objekt table.tr.th Events 1141, 1963
 Objekt table.tr.th Filter 555
 Objekt table.tr.th Styles 846
 Objekt Tag-Bezeichner 247, 482, 562, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 665, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1708
 Objekt Teilzeiger 48
 Objekt text 638



Objekt Text 254, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 1021, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1711, 1776

Objekt Text Abstand zwischen Worten 816, 826, 1949

Objekt Text auf blinkend prüfen 815, 817, 824, 1945

Objekt Text automatischer Zeilenumbruch 816, 826, 1948

Objekt Text dekoratives Layout 815, 824, 1945

Objekt Text durchstreichen 815, 817, 824, 1945

Objekt Text mit eigener Richtung des Renderns 816, 818, 825, 1947

Objekt Text Platz zwischen Worten 816, 826, 1949

Objekt Text Position-Ident in der ersten Textzeile 815, 825, 1946

Objekt Text Rahmen um Text 815, 825, 1946

Objekt Text rendern 816, 818, 825, 1947

Objekt Text überstreichen 815, 818, 824, 825, 1945

Objekt Text unterstreichen 815, 818, 824, 825, 1945

Objekt Text unterstreichen Position des Striches 815, 825, 1947

Objekt Text Wortumbruch bei Überschreitung der Objektgrenzen 816, 826, 1949

Objekt Text Zeilenumbruch automatisch 816, 826, 1948

Objekt Text Zeilenumbruch in Worten 816, 826, 1949

Objekt Textabstand bei asiatischen Zeichen 815, 824, 1945

Objekt textarea 457, 458, 460, 622, 770, 771, 827, 1074, 1082, 1084, 1595, 1607, 1683, 1722, 1733, 1853, 1883

Objekt textarea 1078, 1699

Objekt textarea Events 1142, 1963

Objekt textarea Filter 555

Objekt textarea Styles 846

Objekt Textausrichtung 815, 824, 1944

Objekt Textausrichtung Blocksatz 815, 825, 1946

Objekt Textausrichtung der letzten Zeile 815, 824, 1945

Objekt Textbereich 240, 245, 482, 562, 568, 572, 590, 598, 602, 615, 619, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668

Objekt Textdekoration 815, 817, 818, 824, 825, 1945, 1947

Objekt Textdekoration abschalten 815, 817, 825, 1945

Objekt Textfarbe 810, 820, 1928

Objekt Textfluss asiatische Zeichen 811, 820, 1933

Objekt Textfont 811, 820, 1931

Objekt Textfont Fettheit 811, 820, 1932

Objekt Textfont Höhe 811, 820, 1932

Objekt Textfont Stil 811, 820, 1932

Objekt Textfont Stil-Variante 811, 820, 1932

Objekt Textkonvertierung 815, 825, 1946

Objekt Textkonvertierung jeder Wortanfang mit Grossbuchstabe 815, 825, 1946

Objekt Textkonvertierung nach Großbuchstaben 815, 825, 1946

Objekt Textkonvertierung nach Kleinbuchstaben 815, 825, 1946

Objekt TextNode 280, 458, 1082

Objekt textrange 457, 458, 459, 460, 770, 771, 827, 1083, 1084, 1085, 1582, 1583, 1634, 1711, 1751, 1757, 1764, 1770, 1778, 1828, 1830, 1839, 1842, 1850, 1853, 1883, 1888

Objekt TextRange 280, 458, 1082

Objekt textrange Events 1142, 1963

Objekt textrange Styles 847

Objekt TextRange.TextRectangle 458, 461, 1082, 1085

Objekt textrectangle 1581, 1649, 1682, 1720

Objekt textrectangle Styles 847

Objekt Textumfluss 810, 815, 817, 818, 819, 820, 824, 1927, 1931, 1944

Objekt Textzeichen Abstand 811, 821, 1934

Objekt Textzeichen-Layout-Gitter 811, 820, 1933

Objekt Textzeichen-Layout-Gitter 2D 811, 821, 1933

Objekt Textzeichen-Layout-Gitter Linie 811, 821, 1933

Objekt Textzeichen-Layout-Gitter Typ 811, 821, 1934

Objekt Textzeichen-Layout-Gitter Zeichengröße 811, 820, 1933

Objekt Timeline Typ 709, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 955, 972

Objekt transparent anzeigen 811, 821, 1935

Objekt Transparenz 648, 666, 1564

Objekt typographischer Effekt "Kashida" für Arabisch 815, 825, 1946

Objekt über mehrere Onlinesitzungen Daten speichern 862

Objekt überlappen 816, 826, 1950

Objekt Umfluss 481, 589, 609, 614, 627, 1581

Objekt Umflussrichtung 481, 561, 571, 597, 609, 618, 627, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 739, 743, 753, 774, 1016, 1028, 1032, 1036, 1041, 1046, 1051, 1077, 1191, 1610

Objekt Umgebung 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 714, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685

Objekt und sein Plaintext 458, 1082

Objekt unterstützte HTTP-Methoden 561, 1656

Objekt Url der *.class-Datei 567, 753, 1594

Objekt Url der Daten 601, 1603

Objekt Url der Komponente 567, 753, 1594

Objekt var 99, 149, 160, 1190

Objekt var Events 1142, 1963

Objekt var Filter 555

Objekt var Styles 847

Objekt VBArray 1428, 1429, 1833, 1835

Objekt Verhalten 507, 1761

Objekt Viewbereich 457, 460, 484, 564, 570, 575, 601, 603, 612, 617, 621, 631, 669, 684, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 771, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1085, 1194, 1876

Objekt vordefiniert 68, 106

Objekt vordefiniert zum Browser 220

Objekt vordefinierte Operationen 142

Objekt Vordergrundfarbe 810, 820, 1928

Objekt Wert in einen String umwandeln 148, 212, 1911

Objekt Wert in System-lokale Einstellungen umwandeln 158, 212, 1910

Objekt Wert STYLE-Attribut 810, 817, 820, 1929

Objekt window 281, 383, 388, 419, 1592, 1593, 1594, 1607, 1608, 1609, 1610, 1614, 1623, 1627, 1633, 1651, 1659, 1660, 1662, 1666, 1667, 1682, 1684, 1687, 1699, 1721, 1738, 1741, 1745, 1747, 1748, 1750, 1753, 1757, 1760, 1769, 1775, 1839, 1842, 1844, 1849, 1859, 1872, 1876, 1886, 1892, 1895, 1901, 1903

Objekt window des Internet Explorer 370

Objekt window des Netscape 283



Objekt window Ereignisse 291, 298
 Objekt window erzeugen 437, 1849
 Objekt window Events 1142, 1963
 Objekt window Filter 556
 Objekt window Styles 847
 Objekt window.clientInformation 378, 1592
 Objekt window.clientInformation 412
 Objekt window.clientInformation 412
 Objekt window.clientInformation 1566
 Objekt window.clientInformation 1566
 Objekt window.clientInformation 1567
 Objekt window.clientInformation 1567
 Objekt window.clientInformation 1583
 Objekt window.clientInformation 1598
 Objekt window.clientInformation 1601
 Objekt window.clientInformation 1665
 Objekt window.clientInformation 1672
 Objekt window.clientInformation 1708
 Objekt window.clientInformation 1726
 Objekt window.clientInformation 1727
 Objekt window.clientInformation 1834
 Objekt window.clientInformation 1909
 Objekt window.clientInformation des Internet Explorer 410
 Objekt window.clipboardData des Internet Explorer 413
 Objekt window.dialogArguments 378, 1608
 Objekt window.dialogArguments des Internet Explorer 416
 Objekt window.document des Internet Explorer 418
 Objekt window.document.body des Netscape 312
 Objekt window.document.frame des Netscape 312
 Objekt window.document.frameset des Netscape 318
 Objekt window.document.ilayer des Netscape 326
 Objekt window.document.img des Netscape 324
 Objekt window.document.layer des Netscape 326
 Objekt window.document.link des Netscape 332
 Objekt window.document.span des Netscape 333
 Objekt window.document.style des Netscape 334
 Objekt window.document.styleSheet des Netscape 334
 Objekt window.event des Netscape 353
 Objekt window.external 381, 1162, 1625, 1656, 1737, 1738, 1742, 1827, 1831, 1844, 1901
 Objekt window.history des Netscape 366
 Objekt window.location des Netscape 366
 Objekt window.navigator des Netscape 367
 Objekt window.popup 392, 1172, 1181, 1182, 1614, 1643, 1757, 1899
 Objekt window.screen des Netscape 370
 Objekt Wortanfang mit Grossbuchstabe 815, 825, 1946
 Objekt xml 1195
 Objekt xml Events 1142, 1963
 Objekt xml Styles 847
 Objekt Zeichensatz 424, 560, 736, 749, 1187, 1591
 Objekt Zeiger 48
 Objekt Zeiger auf den Prototyp-Bereich 150, 209, 1677
 Objekt Zeiger auf Kinder des Objektes 268, 271, 467, 471
 Objekt Zeiger auf TextRectangle 460, 483, 564, 569, 574, 591, 600, 602, 611, 616, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1780
 Objekt Zeilenumbruch 811, 821, 1934
 Objekt Zeilenumbruch automatisch 816, 826, 1948
 Objekt zoomen mit Layoutkorrektur der Objektmgebung 816, 826, 1950

Objekt zu dem der Knoten 238
 Objekt Zugriff per Alt + Taste 481, 560, 567, 571, 589, 597, 609, 614, 618, 680, 686, 691, 695, 703, 707, 712, 717, 721, 725, 730, 743, 752, 759, 774, 1014, 1027, 1032, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1190, 1556
 Objekt Zustand des Ladens 681, 686, 703, 1595
 Objektaktivität 561, 572, 597, 609, 614, 618, 627, 666, 681, 686, 691, 696, 700, 703, 708, 713, 717, 721, 725, 743, 765, 774, 885, 893, 897, 904, 908, 914, 921, 926, 937, 945, 949, 954, 965, 971, 1016, 1028, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1191, 1621
 Objektaktivitäten Dauer 885, 893, 897, 904, 908, 914, 918, 921, 926, 931, 936, 940, 945, 949, 954, 971, 1619
 Objekt-Breite 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Objekt-Content Editierbarkeit 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1640
 Objekt-Content-Editierbarkeit 481, 560, 589, 597, 609, 614, 618, 627, 686, 708, 713, 717, 721, 725, 730, 743, 774, 1076, 1191, 1596
 Objekte 52
 Objekte agieren parallel 1713
 Objekte browserintern 224
 Objekte im Browser 131
 Objekte im Dokument 256, 435, 629, 827, 1786
 Objekte im DOM 223
 Objekte im Quellcode 222
 Objekte in Javascript und im Browser 131
 Objekte überlappend 772
 Objektes Fenster-Objekt 648, 666, 1597
 Objekttestatus 239
 Objektfähigkeit des Browsers 136
 Objektframe Events 1135, 1956
 Objektgröße 1128, 1995
 Objekthierarchie 69
 Objekthierarchie logische 222
 Objekthöhe 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1014, 1028, 1032, 1036, 1041, 1046, 1051, 1058, 1063, 1070, 1076, 1187, 1191, 1592
 Objekt-ID 239
 Objektinhalt Mehrzeiligkeit 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 750, 751, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1641
 Objektinstanz 142, 149
 Objekt-Instanz aktuelle 77
 Objektinstanz erzeugen 93
 Objektinstanz löschen 143
 Objektinstanz Objektklasse ermitteln 142
 Objektinstanz Objekttyp ermitteln 142
 Objektinstanz Speicher freigeben 143
 Objekt-Instanz Zeiger 77
 Objektinstanzen vergleichen 143
 Objektklasse 61, 67, 150, 208, 209, 1595, 1596



Objektklasse einer Objektinstanz 142
 Objektklasse eines Objektes prüfen 78
 Objektklasse ermitteln 142
 Objektklasse prüfen 78
 Objektklassen 160
 Objektkonstruktor 111
 Objektname als Literal 148
 Objektname logisch 224
 Objektname logischer 481, 561, 567, 572, 597, 601, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 736, 739, 744, 750, 751, 753, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1195, 1635
 Objektnotation als Literal 148
 Objekttrand unterer Abstand zur Umgebung Oberkante 810, 819, 1927
 Objektreferenzen 796, 976
 Objekt-Sichtbarkeit im Dokument 222
 Objekt-Text liefern 242
 Objekttyp 58, 67, 208
 Objekttyp Basis-Datenstruktur 58
 Objekttyp Bezeichner 61, 150, 209, 1595, 1596
 Objekttyp Datentyp 58
 Objekttyp ermitteln 142
 Objekttypen 160
 objektübergreifende Operationen mit Objekten 142
 objektunabhängige Operationen mit Objekten 142
 Objektvariable 67
 Objektvariable als Array Objekt aus Literalen deklarieren 74
 Objektvariable Deklaration 69
 Objektvariable deklarieren 72
 Objektvariable Erweiterung 70
 Objektvariable Klasse 67
 Objektvariable mit new deklarieren 72
 Objektvariable Typ 67
 Objektvariable von Objektklasse ableiten 69
 Objektzeigerfeld 71
 ObjektZeigerFeld 43, 44
 Objektzeigerfeld dynamisch 71
 Objektzugehörigkeit zum DOM 239, 1825
 Objektzugriff 225
 ObjeT Textfont Familie 811, 820, 1932
 oblique 987
 Oder 75
 oder <LINK> 562, 736, 1679
 Offline-Status der Verbindung 854, 1595
 öffnen Dokument 437, 1849
 öffnen Fenster 395, 437, 1849
 Offscreen 382, 1662
 off-screen bitmap buffer 854, 1182, 1583
 OK-/CANCEL-Button 398, 1859
 OK-Button 388, 1738
 oktal 57
 onabort 1095, 1097, 1112, 1161, 1162, 1210, 1966, 1977
 onAbort 325, 670
 onactivate 1112, 1210, 1966, 1977
 onafterprint 398, 1097, 1112, 1161, 1210, 1859, 1966, 1977
 onafterupdate 1112, 1210, 1966, 1977
 onbeforeactivate 1112, 1210, 1967, 1977
 onbeforecopy 1097, 1112, 1160, 1211, 1967, 1978
 onbeforecut 378, 1097, 1113, 1160, 1211, 1593, 1967, 1978
 onbeforedeactivate 1114, 1211, 1967, 1979
 onbeforeeditfocus 1114, 1211, 1967, 1979

onbeforepaste 378, 1097, 1114, 1160, 1211, 1593, 1967, 1979
 onbeforeprint 398, 1097, 1114, 1161, 1211, 1859, 1968, 1979
 onbeforeunload 1097, 1114, 1161, 1212, 1968, 1979
 onbeforeupdate 1114, 1212, 1968, 1979
 onbegin 887, 895, 899, 900, 906, 910, 911, 916, 917, 919, 924, 929, 930, 933, 934, 948, 951, 956, 957, 967, 973, 974, 1114, 1117, 1126, 1979, 1984, 1994
 onblur 291, 298, 359, 360, 389, 483, 563, 568, 573, 591, 599, 611, 616, 620, 629, 633, 634, 635, 637, 638, 641, 649, 660, 668, 683, 688, 693, 697, 705, 710, 714, 719, 723, 727, 732, 745, 754, 761, 776, 1019, 1030, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1095, 1097, 1115, 1144, 1145, 1193, 1212, 1745, 1968, 1980
 onbounce 1115, 1212, 1968, 1980
 oncellchange 1115, 1212, 1968, 1980
 onchange 634, 641, 1096, 1097, 1212, 1968, 1980
 onclick 483, 484, 563, 565, 568, 570, 573, 575, 591, 593, 599, 601, 611, 612, 616, 618, 620, 622, 629, 631, 633, 634, 637, 638, 683, 684, 688, 690, 693, 695, 698, 699, 702, 705, 707, 710, 711, 715, 716, 719, 720, 723, 725, 727, 729, 732, 734, 740, 742, 746, 747, 754, 756, 761, 763, 767, 769, 776, 778, 1019, 1027, 1030, 1031, 1043, 1045, 1048, 1051, 1054, 1056, 1060, 1062, 1066, 1068, 1072, 1074, 1079, 1082, 1096, 1098, 1107, 1115, 1157, 1159, 1193, 1194, 1212, 1748, 1887, 1968, 1981
 onClick 571, 637, 1598
 oncontentready 503, 1981
 oncontentsave 503, 1981
 oncontextmenu 1098, 1115, 1157, 1212, 1968, 1981
 oncontrolselect 1115, 1212, 1969, 1981
 oncopy 416, 1098, 1115, 1160, 1213, 1781, 1969, 1981
 oncut 416, 1098, 1115, 1160, 1213, 1781, 1969, 1981
 ondataavailable 1116, 1213, 1969, 1982
 ondatasetchanged 1116, 1213, 1969, 1982
 ondatasetcomplete 1116, 1213, 1969, 1982
 ondblclick 484, 565, 570, 575, 593, 601, 612, 618, 622, 631, 684, 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734, 742, 747, 756, 763, 769, 778, 1027, 1031, 1045, 1051, 1056, 1062, 1068, 1074, 1082, 1098, 1107, 1116, 1158, 1159, 1194, 1213, 1887, 1969, 1982
 ondeactivate 1116, 1213, 1970, 1982
 ondetach 503, 1983
 ondocumentready 503, 1983
 ondrag 1098, 1116, 1150, 1214, 1970, 1983
 ondragdrop 291, 298, 359, 360, 1098, 1144, 1145
 ondragend 1099, 1117, 1150, 1214, 1970, 1983
 ondragenter 1099, 1117, 1214, 1970, 1983
 ondragleave 1099, 1117, 1150, 1214, 1970, 1983
 ondragover 1099, 1117, 1214, 1970, 1983
 ondragstart 1099, 1117, 1150, 1214, 1763, 1971, 1983
 ondrop 416, 1099, 1117, 1150, 1215, 1748, 1971, 1983
 onend 887, 895, 899, 900, 906, 911, 917, 919, 924, 929, 930, 933, 934, 948, 951, 956, 957, 967, 973, 974, 1114, 1117, 1127, 1979, 1983, 1994
 onerror 129, 291, 298, 359, 360, 1100, 1117, 1144, 1145, 1147, 1162, 1215, 1737, 1971, 1984
 onError 325, 670
 onerror-Routine privater Art 1147
 onerror-Standard abschalten 1147
 onerrorupdate 1119, 1215, 1971, 1985
 onerror 1096
 onfilterchange 509, 1100, 1105, 1119, 1215, 1971, 1985
 onFilterChange 1105
 onfinish 1119, 1215, 1971, 1985



onfocus 291, 298, 359, 360, 394, 633, 634, 635, 637, 638, 641, 671, 1096, 1100, 1119, 1144, 1145, 1215, 1775, 1971, 1985
onfocusin 1119, 1215, 1971, 1986
onfocusout 1119, 1215, 1972, 1986
onhelp 1100, 1119, 1216, 1972, 1986
onhide 860, 1119, 1986
onkeydown 359, 360, 1100, 1119, 1144, 1145, 1153, 1154, 1155, 1156, 1216, 1972, 1986
onkeypress 356, 359, 360, 1101, 1119, 1144, 1145, 1153, 1154, 1155, 1156, 1216, 1972, 1986
onkeyup 359, 360, 1101, 1120, 1144, 1145, 1155, 1216, 1972, 1986
onlayoutcomplete 1120, 1216, 1972, 1986
Online-Status der Verbindung 854, 1595
Onlinestatus des Browsers 1170, 1665
Online-Zustand des Users 410
onload 292, 298, 359, 360, 850, 858, 862, 863, 865, 1096, 1101, 1120, 1144, 1145, 1161, 1162, 1216, 1973, 1986
onLoad 325, 670
onlosecapture 1101, 1120, 1159, 1216, 1973, 1987
onmediacomplete 887, 908, 911, 914, 917, 919, 921, 924, 927, 930, 931, 934, 945, 948, 954, 956, 957, 971, 973, 974, 1120, 1121, 1655, 1987, 1988
onmediaerror 887, 911, 917, 919, 924, 930, 934, 948, 957, 973, 1120, 1121, 1987
onmousedown 446, 484, 564, 565, 569, 570, 575, 592, 593, 600, 601, 612, 617, 618, 621, 622, 630, 631, 633, 684, 689, 690, 694, 695, 699, 702, 706, 707, 711, 716, 720, 724, 725, 728, 729, 733, 734, 741, 742, 747, 755, 756, 762, 763, 768, 769, 778, 1027, 1031, 1045, 1050, 1051, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1101, 1107, 1121, 1158, 1159, 1194, 1217, 1584, 1865, 1887, 1973, 1988
onmouseenter 1102, 1121, 1158, 1217, 1973, 1988
onmouseleave 1102, 1121, 1158, 1217, 1973, 1988
onmousemove 446, 484, 564, 565, 569, 570, 575, 592, 593, 600, 601, 612, 617, 618, 621, 622, 630, 631, 684, 689, 690, 694, 695, 699, 702, 706, 707, 711, 716, 720, 724, 725, 728, 729, 733, 734, 741, 742, 747, 755, 756, 762, 763, 768, 769, 778, 1027, 1031, 1045, 1050, 1051, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1102, 1107, 1121, 1158, 1159, 1194, 1217, 1584, 1865, 1887, 1973, 1988
onmouseout 446, 484, 564, 565, 569, 570, 575, 592, 593, 600, 601, 612, 617, 618, 621, 622, 630, 631, 684, 689, 690, 694, 695, 699, 702, 706, 707, 711, 712, 716, 720, 724, 725, 728, 729, 733, 734, 741, 742, 747, 755, 756, 762, 763, 768, 769, 778, 1027, 1031, 1045, 1050, 1051, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1096, 1102, 1107, 1121, 1158, 1159, 1194, 1195, 1217, 1865, 1887, 1973, 1988
onMouseOut 571, 1598
onmouseover 292, 298, 359, 360, 446, 484, 564, 565, 569, 570, 575, 592, 593, 600, 601, 612, 617, 618, 621, 622, 630, 631, 684, 689, 690, 694, 695, 699, 702, 706, 707, 711, 712, 716, 720, 724, 725, 728, 729, 733, 734, 741, 742, 747, 755, 756, 762, 763, 768, 769, 778, 1027, 1031, 1045, 1050, 1051, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1096, 1102, 1106, 1107, 1121, 1144, 1145, 1158, 1159, 1194, 1195, 1217, 1865, 1887, 1974, 1988
onMouseOver 571, 1106, 1598
onmouseover Pixelgenauigkeit 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 660, 663, 668, 683,

688, 693, 698, 701, 705, 710, 715, 719, 723, 727, 732, 737, 740, 746, 751, 755, 761, 767, 776, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1189, 1193, 1196, 1751
onmouseup 446, 484, 564, 565, 569, 570, 575, 592, 593, 600, 601, 612, 617, 618, 621, 622, 630, 631, 633, 684, 689, 690, 694, 695, 699, 702, 706, 707, 711, 716, 720, 724, 725, 728, 729, 733, 734, 741, 742, 747, 755, 756, 762, 763, 768, 769, 778, 1027, 1031, 1045, 1050, 1051, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1102, 1107, 1121, 1158, 1159, 1194, 1218, 1584, 1865, 1887, 1974, 1988
onmousewheel 1107, 1121, 1218, 1733, 1974, 1988
onmove 292, 298, 359, 360, 1103, 1122, 1144, 1145, 1218, 1974, 1989
onmoveend 1122, 1218, 1974, 1990
onmovestart 1123, 1218, 1975, 1990
onopenstatechange 860, 1124, 1991
onoutofsync 911, 917, 924, 930, 934, 948, 957, 974, 1124, 1991
onpaste 1103, 1124, 1160, 1219, 1975, 1991
onpause 887, 895, 900, 906, 911, 917, 924, 930, 934, 948, 951, 957, 967, 974, 1125, 1992
onplaystatechange 860, 1126, 1993
onpropertychange 502, 503, 1078, 1103, 1105, 1107, 1126, 1219, 1676, 1699, 1772, 1975, 1993
onPropertyChange 1105
onreadystatechange 504, 1126, 1219, 1975, 1993
onrepeat 887, 892, 895, 897, 899, 900, 903, 906, 911, 917, 919, 924, 929, 930, 933, 934, 948, 951, 956, 957, 964, 967, 973, 974, 1114, 1117, 1126, 1556, 1979, 1984, 1994
onreset 622, 626, 630, 637, 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 974, 1096, 1103, 1127, 1219, 1870, 1975, 1994
onresize 292, 298, 359, 360, 1103, 1128, 1144, 1145, 1219, 1975, 1995
onResize 1166, 1824, 1995
onresizeend 1103, 1128, 1219, 1975, 1995
onresizeend 1103, 1128, 1219, 1976, 1995
onresume 887, 895, 900, 906, 911, 917, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1995
onreverse 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1996
onrowenter 1129, 1220, 1976, 1996
onrowexit 1129, 1220, 1976, 1996
onrowsdelete 1129, 1220, 1976, 1996
onrowsinserted 1129, 1220, 1976, 1996
onsave 862, 863, 864, 1129, 1996
onscroll 1103, 1129, 1220, 1976, 1996
onseek 887, 911, 917, 920, 924, 930, 934, 949, 957, 967, 974, 1129, 1996
onselect 641, 1096, 1104, 1130, 1220, 1976, 1998
onselectionchange 1130, 1220, 1976, 1998
onselectstart 1130, 1220, 1976, 1998
onshow 860, 1130, 1998
onstart 743, 747, 1130, 1220, 1906, 1977, 1998
onstop 743, 748, 1104, 1130, 1161, 1220, 1907, 1977, 1998
onsubmit 622, 626, 631, 638, 1097, 1104, 1131, 1221, 1908, 1977, 1999
onsyncstored 911, 917, 924, 930, 934, 948, 949, 957, 974, 1124, 1131, 1991, 1999
ontimeerror 1131, 1221, 1977, 1999
ontrackchange 917, 930, 949, 974, 1131, 1999



onunload 292, 298, 359, 360, 1097, 1104, 1131, 1144,
 1145, 1161, 1162, 1221, 1977, 1999
 onURLFlip 917, 928, 930, 972, 1107, 1131, 1726, 2000
 onXXX-Ereignis-Handler überschreiben 291
 open(["mime-typ"] [, "replace"]) 437
 opener 283
 Opera-Browser 1167
 Operationen mit Ketten 214
 Operationen mit Objekten 142
 Operationen mit Objektinstanz 142
 Operationen mit String-Literalen 214
 Operationen mit Strings 214
 Operator !== 78
 Operator % 78
 Operator ?: 66, 80
 Operator ?: 91
 Operator << 79
 Operator === 78
 Operator >> 80
 Operator >>> 80
 Operator delete 78, 80
 Operator der Verkettung 214
 Operator für Zuweisung 76
 Operator in 91
 Operator instanceof 78, 81
 Operator new 78, 93
 Operator this 77
 Operator typeof 54, 77, 78, 81, 142
 Operator valueOf 143
 Operator void 77, 78, 81
 Operator with 77
 Operator zur Bitverschiebung 78
 Operatoren 75
 Operatoren arithmetische 75
 Operatoren bitweise 76
 Operatoren für Vergleich 76
 Operatoren für Verkettung von Zeichenketten 76
 Operatoren in Javascrpt 1.5 83
 Operatoren in Microsoft JScript 77
 Operatoren logische 75
 Operatoren Prioritäten 82
 Operatoren Standard-Prioritäten 82
 OPTION 757, 763, 767, 1299, 1711
 Option aus einer Selektion 763
 Option innerhalb einer Optionsgruppe 760, 1687
 Option Label 766, 1649
 Option laufende Nummer 765, 1635
 Option laufende Nummer in List-Box 765, 1635
 option objekt 248, 463, 465, 1727
 option Objekt 842
 option Objekt Events 1139, 1960
 option Objekt Filter 553
 Optionen 456, 757, 763, 769
 Optionindex im select Objekt 760, 1687
 Optionsgruppe 760, 1687
 Ordner Attribute 1439, 1570
 Ordner erzeugen 1431, 1757
 Ordner kopieren 1431, 1441, 1753
 Ordner löschen 1432, 1441, 1758, 1759
 Ordner Recent 1445
 Ordner verschieben 1433, 1441, 1839, 1840
 Ordneränderung Datum 1439, 1606
 Ordnererstellung Datum 1439, 1605
 Ordnergröße in Bytes 1441, 1692
 Ordnen des Betriebssystems 1433, 1802
 Ordnername 1440, 1660

Ordnername als 8.3 Bezeichner 1440, 1689
 Ordnerpfad 1440, 1671
 Ordnerpfad als 8.3 Pfad 1440, 1689
 Ordnersuffix 1441, 1724
 Ordnerzugriff Datum 1439, 1606
 orphans 987
 Other 1601
 out 1657
 Outdent 278, 477
 outerHeight 439
 outerWidth 439
 outset 983, 987, 991
 overline 987, 990
 OverWrite 278, 477
 P 1301
 Padding 589, 605, 690, 774, 796, 803, 811, 812, 813,
 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721,
 1935, 1936, 1938, 1939
 page Objekt 818, 996, 998, 1553, 1678, 1687, 1738,
 1917
 pageDown 483, 1080, 1763
 Page-Enter 509, 749
 Page-Exit 509, 749
 pageLeft 483, 1080, 1763
 pageRight 483, 1080, 1763
 pageUp 483, 1080, 1763
 par 1713
 PAR 869
 parallel 869
 PARAM 1305
 Parameter 44, 72
 Parameter Anzahl 104, 160
 Parameter einer Funktion 103, 195, 1650
 Parameter und Parameterliste 104
 Parameter zum Eventhandler 1143, 1147
 Parameterliste Funktion 88, 106
 Parameterversorgung der Methoden 223
 Parameterliste im Funktionsaufruf 104
 parent 283
 Parent 222
 parseFloat() 147, 157, 200, 1852
 parseFloat(navigator.appVersion) 411, 1167, 1567
 parsen 222, 1188, 1723
 Parsen bedingt 53, 99
 Parsen und Aufwand für Browser 48, 61
 Parsen und Aufwand für Scriptmaschine 48, 61
 parsen und Punktnotation 48, 61
 Pars-Reihenfolge des Internet Explorer 605, 774, 807
 Pars-Status des Scriptes 1187, 1607
 password 635, 685, 1722
 Password 707
 PASSWORD 1096
 Password als Cookie 485
 Password permanent speichern 1163, 1742
 Password speichern 1163, 1742
 Password-Control 1722
 Passwordeingabe aktivieren 635
 Passwordeingabe deaktivieren 635
 Password-Feld 1572
 Password-Zugriffschutz 51
 Paste 278, 477
 path= 488
 pathname 735
 PATH-Variable 1460
 pausieren auf der Timeline beenden 1873
 pausieren auf Timeline 592, 1853



pausieren in der Timeline 1645
 PEERS 941
 Performance Browser 48, 61, 876
 periodischer Aufruf eines Scripts 401, 1892
 permanenter Fenstertitel-Wechsel 428, 1714
 personalbar 439
 persönliche Toolbar 439
 Pfad eines Objektes 562, 572, 1167, 1671
 Pfadangabe lokal 56, 58, 114
 Pfade für Dateien 24
 Pfadtrenner 49
 Pfeil Nord 988
 Pfeil Nord-Ost 988
 Pfeil Nord-West 988
 Pfeil Ost 988
 Pfeil Süd 988
 Pfeil Süd-Ost 988
 Pfeil Süd-West 988
 Pfeil West 988
 physische Window-Name 437
 Pi 199, 1672
 PI 199, 1672
 Pixelate Filter 537
 Pixelpos Objekt 434, 1764
 Pixel-Position der Maus 1104
 Pixelposition des Rechteckes um Objekt 462, 1087,
 1581, 1649, 1682, 1720
 Pixelpositions-Angaben 984
 Plain-Text 224, 239, 242, 258, 261, 484, 564, 570, 574,
 575, 592, 600, 601, 603, 612, 617, 621, 630, 651, 661,
 664, 669, 684, 689, 690, 694, 698, 699, 701, 702, 706,
 711, 715, 716, 720, 724, 728, 729, 733, 738, 741, 746,
 747, 756, 762, 763, 768, 777, 778, 827, 1016, 1017,
 1027, 1028, 1029, 1031, 1032, 1033, 1035, 1036, 1037,
 1039, 1041, 1042, 1045, 1047, 1050, 1052, 1055, 1058,
 1059, 1061, 1064, 1065, 1067, 1068, 1070, 1071, 1073,
 1081, 1190, 1191, 1194, 1636, 1666, 1829, 1869
 Plaintext eines Objektes 458, 1082
 Plain-Text im Objekt 482, 561, 568, 572, 598, 601, 610,
 615, 619, 628, 663, 667, 682, 687, 692, 696, 700, 704,
 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775,
 1000, 1017, 1042, 1047, 1052, 1059, 1065, 1071, 1077,
 1192, 1667
 Plain-Text im Textbereich 459, 1083, 1711
 Plain-Text im Textbereiches dehnen 459, 1084, 1770
 Plain-Textdaten 280
 Plain-Textelement 253, 280, 1757
 Plain-Textelement im Dokument 253, 280, 1757
 platform 1167
 Plattform 410
 Plattform Browser 1567
 Plattform und Browserversion 1169, 1567
 Platz zwischen Worten 816, 826, 1949
 Play(); 528, 672
 PlayCount 874, 1469
 playItem Objekt 911
 playItem Objekt aktiv 939, 1561
 playItem Objekt aktivieren 928, 929, 932, 933, 940, 947,
 955, 1847, 1857
 playItem Objekt Index in der Collection playList 1635
 playList Collection 911, 913, 914, 925, 926, 927, 940,
 944, 945, 970, 971, 1575, 1576, 1578, 1632, 1635
 Playlist von CD 874
 Playliste 860, 861, 875, 911, 1470
 Playlisten-Datei 939, 1561
 PlaylistInfo Objekt 859, 860, 861, 1631, 1660

Plugin Adobe Acrobat 369, 1169, 1170
 Plugin Beschreibung 369, 1169, 1170
 Plugin Daten übergeben 369, 1169, 1170
 Plugin -Ersatz durch ActiveX-Control beim IE 451, 613,
 756, 1171
 Plugin laden 369, 1168
 Plugin mit Daten versorgen 451, 613, 756, 1171
 Plugin neu laden 369, 1168
 Plugin und erlaubte Dateisuffixe 369, 1169, 1171
 Plugin und sein Mimetype 369, 1169, 1170
 Plugin-Dateiname 368, 1168
 plugin-eigene Eigenschaften 369, 1169, 1170
 plugin-eigene Eigenschaften 451, 613, 756, 1171
 plugin-eigene Methoden 369, 451, 613, 756, 1169, 1170,
 1171
 plugin-fähiger Browser 1171
 Plugin-Kurzbeschreibung 368, 1168
 Pluginname 368
 Plugins ab IE 6.x 369, 1169, 1170
 Plugins auflisten 368, 1168
 Plugins beim Netscape 368, 1168
 plugins Collection 451, 613, 756
 Plugins des Netscape 1375
 plugins.length 1168
 plugins[.description] 1168
 plugins[.filename] 1168
 plugins[.name] 1168
 Pluginsanzahl 368, 1168
 PLUGINSOURCE 309, 451, 613, 756
 PNG 671, 703
 pointer 988
 poly 571, 1598, 1688
 polygon 1688
 popup Objekt Events 1139, 1960
 popup Objekt Styles 842
 Pop-up-Fenster 392, 1178, 1757
 Pop-up-Fenster 1172
 Pop-up-Fenster anzeigen 1182, 1899
 Pop-up-Fenster Anzeigezustand 1181, 1643
 Pop-up-Fenster HTML-Dokument 1181, 1614
 Pop-up-Fenster schliessen 1181, 1825
 Pop-uphilfe 981
 Pop-uphilfen 981
 port 735
 Port 561, 572, 1167, 1633
 Port 21 562, 572, 1167, 1675
 Port 80 562, 572, 1167, 1675
 Portable Network Graphics 671, 703
 Position der zu ladenden Seite innerhalb der History
 1165
 Position der Maus 1104
 Position der Maus im HTML-Element 1105
 Position sichtbarer Bereich über dem Objekt 810, 820,
 1928
 Positionen von 2 Knoten im DOM tauschen 266, 565,
 570, 575, 593, 601, 603, 613, 618, 622, 632, 651, 662,
 665, 670, 684, 690, 695, 699, 702, 707, 712, 716, 721,
 725, 729, 734, 738, 742, 748, 756, 763, 769, 778,
 1027, 1032, 1035, 1039, 1045, 1051, 1056, 1062, 1068,
 1074, 1082, 1190, 1195, 1909
 Positionierung absolut 277, 476
 Positionsangaben als bezüglich Anzeigefenster-Rand
 984
 Positionsangaben als bezüglich Vorgänger-Element 984
 positiv unendlich 205
 POSITIVE_INFINITY 205, 1675



Positiv-Unendlich 1675
 post 622, 1656
 Potenz 199, 1855
 PPC 411, 1601
 pre 990
 preference() 1168
 previous 1165
 print 1655
 Print 278, 477
 Prinzipien der Eventbehandlung des Netscape 359
 Prioritäten Operatoren 82
 priorityClass Objekt 918
 private Ausnahmebedingung 192
 private Datentypen 54
 private Funktion 106
 privater Konstruktor 61, 150, 209, 1595, 1596
 privater Run-Time-Error 194, 1659
 privates Objekt 68
 privates Tag 504
 Privileg 283, 361, 1145, 1146
 Privilegänderung 361, 1146
 Privilegmanager des Netscape 283, 361, 1146
 Privilegmanager 361, 1145
 PROCEDURE 106
 Programme und Dateien in einem IFRAME starten 665, 1685
 Programmierer 223
 Programmierung Timer 866
 Programmierung von Timern 865
 protocol 735
 Protokoll 562, 572, 1166, 1167, 1675
 Protokoll SMTP 563, 737, 754, 1721
 Protokoll-Teil einer Url inklusive http und ftp liefern 428, 562, 572, 682, 1677
 Prototyp-Bereich Objekt 150, 209, 1677
 Prototyping 44, 66, 70, 71, 114, 150, 209, 223, 676, 1677
 Prototyping durch Funktion 103
 Prototyping einer Methode 103
 Prototyping JScript-Objekt 150, 209, 1677
 Prototyping Objekt 150, 160, 209, 1677
 Prototyping Script-Objekt 160
 Protoyping 111
 Proxyserver-Cache 748, 1596, 1635
 prüfen auf ActiveX 1375
 prüfen Event-Eigenschaften 1143
 prüfen Textbereiche 460, 1084, 1828, 1830
 Pseudoklasse 797, 847
 Pseudoklasse
 first 818
 Pseudoklasse
 left 818
 Pseudoklasse
 right 818
 Pseudoklasse
 first 998
 Pseudoklasse
 left 998
 Pseudoklasse
 right 998
 Pseudoklasse
 first 1553
 Pseudoklasse
 left 1553
 Pseudoklasse
 right 1553

Pseudoklasse
 first 1917
 Pseudoklasse
 left 1917
 Pseudoklasse
 right 1917
 Pseudoklasse @page 998
 Pseudoklasse des a Objektes für aktiven Link 809, 818, 1917
 Pseudoklasse des a Objektes für Link, der bereits aktiviert wurde 809, 818, 1917
 Pseudoklasse des a Objektes für Link, der nicht aktiviert wurde 809, 818, 1917
 Pseudoklasse des a Objektes für Link, der nicht kürzlich aktiviert wurde 809, 818, 1917
 Pseudoklasse für Style der ersten Zeile 809, 818, 1917
 Pseudoklasse für Style des ersten Buchstaben 809, 818, 1917
 Pseudoklassen 830, 842, 849
 public 450, 566, 570
 PUBLIC 802
 PUBLIC:ATTACH 492
 PUBLIC:COMPONENT 493
 PUBLIC:DEFAULTS 494
 PUBLIC:EVENT 495
 PUBLIC:METHOD 497
 PUBLIC:PROPERTY 497
 Pull-Down 397, 439
 Punkt 75, 980
 Punkt für Tausendertrenner 1376
 Punktnotation 451, 613, 756, 1171
 Punktnotation berechnen 48, 61
 Punktnotation und Browserperformance 48, 61
 Punktnotation und parsen 48, 61
 Punktnotation und Scriptmaschine 48, 61
 Punktnotation und Zeigerbildung 48, 61
 Punktnotation verkürzen 48, 61
 pushWipe 1704, 1723
 Quadratwurzel 199, 1906
 Quadratwurzel von 0,5 199, 1694
 Quadratwurzel von 2 199, 1694
 Quellcode 222, 736
 Quellcode Übersichtlichkeit 48, 61
 Quelltextzeile 295, 396, 438
 Querystring 562, 572, 1167, 1685
 Queue 854
 RadialWipe Filter 517, 538, 2007
 radio 635, 685, 1722
 RADIO 1096
 Radio Button-Control Selektionsstatus 691, 712, 713, 1591, 1606
 Radiobox 1097
 Radiobutton 636
 Radio-Button-Control 712, 1722
 Radius 571, 1599
 Rahmen 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 1076, 1187, 1191, 1592
 Rahmen Dokumentfenster 981
 Rahmen Objekt Art alle 4 Seiten 810, 819, 1925
 Rahmen Objekt Art links 809, 819, 1923
 Rahmen Objekt Art oben 810, 819, 1926
 Rahmen Objekt Art rechts 810, 819, 1924
 Rahmen Objekt Art und Dicke alle 4 Seiten 809, 819, 1920
 Rahmen Objekt Art und Dicke links 809, 819, 1922



Rahmen Objekt Art und Dicke oben 810, 819, 1925
 Rahmen Objekt Art und Dicke rechts 810, 819, 1924
 Rahmen Objekt Art und Dicke unten 809, 819, 1921
 Rahmen Objekt Art unten 809, 819, 1921
 Rahmen Objekt Dicke alle 4 Seiten 810, 819, 1927
 Rahmen Objekt Dicke links 809, 819, 1923
 Rahmen Objekt Dicke oben 810, 819, 1926
 Rahmen Objekt Dicke rechts 810, 819, 1925
 Rahmen Objekt Dicke unten 809, 819, 1922
 Rahmen Objekt Farbe alle 4 Seiten 809, 819, 1922
 Rahmen Objekt Farbe links 809, 819, 1923
 Rahmen Objekt Farbe oben 810, 819, 1926
 Rahmen Objekt Farbe rechts 810, 819, 1924
 Rahmen Objekt Farbe unten 809, 819, 1921
 Rahmen Objekt Singleborder 809, 819, 1922
 Rahmen um eine Tabelle 1016, 1627
 Rahmen um Text 815, 825, 1946
 Rahmendicke 659, 666, 681, 1014, 1581
 Rahmen-Eigenschaften 983
 Rahmenfarbe 648, 659, 983, 1014, 1057, 1063, 1069, 1581
 RandomBars Filter 539
 RandomDissolve Filter 540
 RangeError Run-Time-Error 194, 1659
 Rating der Media-Datei 915, 922, 927, 937, 945, 971, 1678
 Recent-Ordner 1445
 rechte Maustaste 1102, 1106
 rechte Maustaste gedrückt 1104
 rechte oder linke Maustaste 1101
 rechte untere Ecke 993
 rechte untere Ecke des Objektes 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1037, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1192, 1664
 rechte unteren Ecke des Objektes 482, 561, 567, 572, 598, 609, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1191, 1662
 Rechteck um Objekt Pixelposition 462, 1087, 1581, 1649, 1682, 1720
 Rechteck um Textbereich 459, 1083, 1582, 1583
 rechtsbündig 986, 990
 recordset 1379
 recordset.MoveFirst() 1379
 recordset.MoveNext() 1379
 recordset.MovePrevious() 1379
 rect 571, 1598, 1688
 rectangle 1688, 1704, 1723
 Rectangle 460, 564, 569, 574, 629, 683, 689, 693, 698, 705, 710, 715, 719, 724, 728, 732, 737, 741, 746, 755, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1061, 1067, 1072, 1080, 1084, 1193, 1780
 ReferenceError Run-Time-Error 194, 1659
 Referenz als absoluter Pfad 48
 Referenz auf Feld der Zeiger auf TextRectangle-Objekte 460, 483, 564, 569, 574, 591, 600, 602, 611, 616, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1780
 Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag 482, 561, 567, 572, 598, 610, 615, 619, 628, 659, 663, 665, 667, 682, 687, 692, 696, 700, 704,

709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1029, 1033, 1037, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1666
 Referenz auf das document Objekt des Knoten 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1667
 Referenz auf das document-Objekt 239
 Referenz auf das Elternobjekt Body 482, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1195, 1667
 Referenz auf das ERSTE Kind 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739, 744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 Referenz auf das LETZTE Kind 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Referenz auf das NACHFOLGENDE Kind 243, 561, 567, 572, 590, 598, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1660
 Referenz auf das Vorgängerkind 246, 562, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 760, 766, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1077, 1192, 1675
 Referenz auf das Vorgängerkind Body 590, 615, 760, 766
 Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag 481, 561, 597, 609, 618, 627, 659, 662, 665, 666, 730, 739, 744, 775, 1000, 1016, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1636, 1637
 Referenz auf den gesamten Plain-Text im Objekt 482, 561, 568, 572, 598, 601, 610, 615, 619, 628, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 740, 744, 753, 760, 775, 1000, 1017, 1042, 1047, 1052, 1059, 1065, 1071, 1077, 1192, 1667
 Referenz auf den obersten Knoten des XSL-Dokumentes 249, 430, 1734
 Referenz auf Elternknoten 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668
 Referenz auf Feld aller Elemente mit gemeinsamer URN 271, 274, 452, 453, 455, 456, 471, 474, 632, 641, 642, 685, 707, 738, 742, 763, 770, 1000, 1040, 1046, 1190, 1913
 Referenz auf Feldelement 452, 453, 454, 455, 456, 457, 461, 630, 641, 642, 652, 685, 707, 738, 742, 770, 771, 998, 1000, 1040, 1046, 1085, 1171, 1190, 1842
 Referenz auf HTML-Tag 61
 Referenz auf Objekt anhand ID 239



Referenz auf Objekt anhand NAME 239
 Referenz auf Objekt anhand Tag-Name 239
 Referenz auf TextRectangle 591, 616, 762, 768
 Referenz auf TextRectangle-Objekt 460, 483, 564, 569, 574, 600, 602, 611, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1778
 Referenz auf Wurzelknoten (root node) des Dokumentes 243, 1616
 Referenz auf XML-Dokument 249, 430, 863, 864, 1188, 1195, 1734
 Referenz der Eltern 482, 561, 567, 572, 598, 601, 610, 619, 628, 649, 667, 681, 687, 692, 696, 700, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1070, 1077, 1191, 1663
 Referenz der Eltern Body 590, 615, 760, 766
 Referenz Filter 509
 Referenz und Zeiger 48
 Referenzierung einer Variable 64
 Referenzierung Objekt 582
 Refresh-Intervall des Bildschirmes 1183, 1725
 Refresh 278, 477, 1635
 Refresh Dokument 369, 1168
 RegExp JScript-Objekt 149
 regexp Objekt 114, 1183
 RegExp Objekt 114, 149, 160, 218, 1869
 RegExp Script-Objekt 160, 1186
 RegExpError Run-Time-Error 194, 1659
 regionale Sprache des Betriebssystems 1170, 1727
 regionaler Standardzeichensatz 425, 1606
 registrieren eines Events 433, 483, 507, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 660, 664, 668, 683, 688, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 1019, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1079, 1107, 1189, 1193, 1759
 registrieren eines Events 388, 393, 430, 482, 507, 563, 568, 573, 591, 599, 602, 610, 616, 620, 629, 649, 660, 663, 668, 683, 688, 693, 697, 701, 705, 710, 714, 719, 723, 727, 732, 737, 740, 745, 754, 761, 767, 776, 1019, 1029, 1034, 1038, 1043, 1048, 1053, 1060, 1066, 1072, 1079, 1107, 1189, 1193, 1740, 1741, 1759
 Registry-Eintrag 1163, 1742
 Rekursion 112
 Rekursion einer Funktion 161, 1589
 Rekursion mit Timer 401, 1892
 Rekursion und document.write() im HEAD 113
 Rekursion und document.writeln() im HEAD 113
 Rekursion und Ereignisüberwachung 401, 404, 1892, 1895
 Rekursion und Eventüberwachung 401, 404, 1892, 1895
 Rekursion und Variablenübergabe 113
 REL 736, 994, 995, 997
 REL= 420, 982, 1679
 relative 984
 relative Pfade 24
 releaseCapture() 1101, 1147
 releaseEvents() 360, 1145
 Reload des Dokumentes nach Resize des Browserfenster IE 1166, 1824, 1995
 Reload Dokument 772
 Reload eines Dokumentes mit allen seinen FRAMES 315, 321, 644, 655
 Reload mit allen FRAMES 315, 321, 644, 655

reload() 315, 321, 645, 655
 reload([true]) 1166
 RemoveFormat 278, 477
 rendern horizontal 810, 820, 1930
 rendern von Objekten 223
 replace(url) 1166
 reset 637, 685, 1721, 1722
 Reset 1096, 1725
 RESET 1096
 reset Formular 630, 1870
 Reset wird ausgeführt 1096
 Reset wird nicht ausgeführt 1096
 Resetaktion abfangen 1103, 1104
 Reset-Button 1103, 1721
 Reset-Button für Formular 716
 Reset-Button Label 716
 Reset-Button Titel 716
 resizable 397, 399, 406, 408, 439, 1872
 restricted 1685
 Restricted 974
 return 1096, 1107, 1681
 return Anweisung 95, 105, 106
 return false 1096
 return false; 360, 1144
 return true 1096
 return true; 1105
 Returnwert für den Eventhandler 1107, 1681
 RevealTrans Filter 540
 Revisit-After 748
 RFC1766 561, 736, 1634
 rgb(rrr,ggg,bbb) 987, 990, 991, 992
 RGB-Farbangaben 805
 rhs 1016, 1627
 ridge 983, 987, 991
 riesig 987, 989
 right 483, 986, 990, 1080, 1562, 1610, 1763
 Right Margin Body 590
 Robots 748
 robots.txt als Zeilenfolge-Script erstellen 49
 robots.txt die nicht auf dem Server vorhanden ist 49
 robots.txt kann entfallen 50
 robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen gescannt werden 50
 robots.txt und ab Zeile 2: Alle Daten auf dem Server dürfen NICHT gescannt werden 50
 robots.txt und ab Zeile 2: Bestimmte Daten auf dem Server dürfen NICHT gescannt werden 50
 robots.txt und ab Zeile 2: Bestimmte HTML-Dateien auf dem Server dürfen NICHT gescannt werden 50
 robots.txt und ab Zeile 2: Bestimmte Verzeichnisse auf dem Server dürfen NICHT gescannt werden 50
 robots.txt und ab Zeile 2: NUR bestimmte Daten auf dem Server dürfen gescannt werden 50
 robots.txt und Aufbau 49
 robots.txt und Beispiele 50
 robots.txt und ihre Blöcke 49
 robots.txt und ihre Lage auf dem Server 49
 robots.txt und Suchmaschinen 49
 robots.txt und Zeile 1: ALLE Suchmaschinen dürfen scannen 49
 robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf NICHT scannen 49
 robots.txt und Zeile 1: Eine bestimmte Suchmaschinen darf scannen 49
 robots.txt und Zeile 1: KEINE Suchmaschine darf scannen 49



robots.txt-Block: Aufbau ab Zeile 2 49
 robots.txt-Block: Aufbau Zeile 1 49
 Root 1432, 1435, 1440, 1646, 1682, 1776
 Root der Webseite 24
 Root Node 425
 Root Node des Dokumentes 243, 1616
 Rot-Anteil 980
 row 1065, 1071, 1683
 rowgroup 1065, 1071, 1683
 rows 1017, 1683
 rtl 481, 849, 1581, 1610
 Rückkercode des Eventhandlers 1105, 1159
 Rückschritt 56, 58, 113, 114
 Rücksetzen des Formulars 626
 rücksetzen eines Formulars 1103
 Rücksprung zur Startseite 749
 Rumpf der Funktion 88, 106
 Rumpf des Dokumentes 584
 Rumpf des HTML-Dokumentes 584
 runden 199, 1874
 Runden auf n Nachkommastellen 199, 204
 Runtime Error 129
 Runtime Fehler abfangen 128
 Runtime Fehler JScript 115
 Runtime Fehler Microsoft JScript 115
 Runtime-Error 129, 1100, 1147
 Run-Time-Error Ausnahmetyp 194, 1659
 Run-Time-Error Beschreibung 193, 1607, 1656
 Run-Time-Error ConversionError 194, 1659
 Run-Time-Error Nummer 194, 1662
 Run-Time-Error privat 194, 1659
 Run-Time-Error RangeError 194, 1659
 Run-Time-Error ReferenceError 194, 1659
 Run-Time-Error RegExpError 194, 1659
 Run-Time-Error Typ 194, 1659
 Run-Time-Error TypeError 194, 1659
 Run-Time-Error URIError 194, 1659
 runtimeStyle Objekt 849
 runtimeStyle Objekt Filter 553
 runtimeStyle Objekt Styles 842
 rzeichenfolge 1184
 SAMI 1484
 SAMI-Datei 1484
 Sammlungen 223
 Sanduhr 988
 SaveAs 278, 477
 savePreferences() 1168
 Scan-Standard für Suchmaschinen 49
 Scanvorgänge von Suchmaschinen 49
 Schatten bei 3D-Element 981
 Schatten eines Bildes 542
 SCHEME 748
 schliessen Browserfenster 1104
 schliessen Fenster 391, 397, 439, 1750
 schliessen nach Wartezeit 292, 392, 1750
 Schlüssel der Datenspeicherung 1427
 Schriftdatei laden 981
 Schriften 980
 Schusterjunge 987
 SCR 325, 670
 screen 1313, 1655
 screen Objekt 281, 283, 370, 383, 1182, 1183, 1574,
 1583, 1594, 1607, 1608, 1632, 1652, 1684, 1725, 1733
 screen Objekt Styles 842
 screenX 440
 screenY 440

Script 1101
 SCRIPT 751, 1313
 Script Download 1188, 1607
 Script ECMA 1723
 Script im Dokument 1190
 Script JavaScript 1723
 Script JScript 1723
 script language 30, 39
 Script Mimetyp 1188, 1723
 script Objekt 1187, 1188, 1190, 1607, 1623, 1634
 script Objekt Events 1139, 1960
 script Objekt Styles 842
 Script Pars-Status 1187, 1607
 Script per Timer einmalig aufrufen 404, 1895
 Script per Timer periodisch aufrufen 401, 1892
 Script periodisch aufrufen 401, 1892
 Script signiert 101, 283
 Script sofort ausführen 394, 1769
 Script Speichermangel 114
 Script Sprache 1187
 Script Status der Parsens 1187, 1607
 Script und Browserperformance 48
 Script und Ereignishandler 1188, 1634
 Script und Eventbehandlung 1188, 1623, 1634
 Script VB-Script 1723
 Script XML 1723
 Scriptcode 1187
 Scriptcode ausführen 211
 Scriptcode dynamisch erzeugen 211
 Script-Code einfügen 240, 258, 484, 564, 574, 592, 600,
 603, 612, 617, 621, 630, 661, 669, 684, 689, 694, 698,
 701, 706, 711, 715, 720, 724, 728, 733, 741, 746, 762,
 768, 777, 827, 1031, 1067, 1073, 1081, 1194, 1829
 Scriptcode Übersichtlichkeit 48, 61
 Script-Debugger 130, 1100, 1147
 Scriptfehler 66, 114
 Scriptfehler wegen Speichermangel 66
 Scripting.FileSystemObject 1447
 Scriptkommando 928, 972
 Scriptlet 1287
 Scriptmaschine 17, 52, 106, 160, 1188, 1723
 Scriptmaschine Abwärtskompatibilität 18, 31, 42, 135
 Scriptmaschine Buildnummer 100
 Scriptmaschine Buildnummer 40
 Scriptmaschine Buildnummer 1875
 Scriptmaschine des Internet Explorers 18, 31, 42, 135
 Scriptmaschine Hauptversion 40, 1875
 Scriptmaschine Sprache 40, 1875
 Scriptmaschine und Aufwand beim Parsen 48, 61
 Scriptmaschine und Browser 48, 61
 Scriptmaschine und Punktnotation 48, 61
 Scriptmaschine unter Windows 98 18, 31, 42, 135
 Scriptmaschine unter Windows XP 18, 31, 42, 135
 Scriptmaschine Unterversion 40, 1876
 Scriptmaschine Version 100
 Scriptmaschine Versionen 17
 Script-Objekt arguments 104, 160, 1567, 1589, 1650,
 1739, 1745
 Script-Objekt Array 160
 Script-Objekt Array 54
 Script-Objekt Array 162
 Script-Objekt Array 166
 Script-Objekt Array 166
 Script-Objekt Array 167
 Script-Objekt Array 167
 Script-Objekt Array 168



Script-Objekt Array 168
 Script-Objekt Array 169
 Script-Objekt Array 172
 Script-Objekt Array 188
 Script-Objekt Array 1650
 Script-Objekt Array 1752
 Script-Objekt Array 1834
 Script-Objekt Array 1874
 Script-Objekt Array 1905
 Script-Objekt Array 1906
 Script-Objekt Array 1906
 Script-Objekt Array Anzahl der Elemente 166, 1650
 Script-Objekt Array aus Literalen 174
 Script-Objekt Array Elemente anhängen 166, 1752
 Script-Objekt Array Feldelemente physisch sortieren 168, 1906
 Script-Objekt Array komplett nach String konvertieren 167, 1834
 Script-Objekt Array Reihenfolge der Feldelemente physisch umkehren 167, 1874
 Script-Objekt Boolean 56, 160, 175
 Script-Objekt Date 56, 160, 176, 1781, 1783, 1788, 1791, 1793, 1795, 1797, 1800, 1804, 1807, 1809, 1811, 1813, 1815, 1818, 1820, 1822, 1851, 1888, 1892, 1894, 1895, 1898, 1899, 1909, 1910, 1911, 1912, 1913
 Script-Objekt Date Datum-Operation 180, 181, 182, 183, 184, 185, 186, 187, 1781, 1783, 1788, 1791, 1793, 1795, 1797, 1800, 1802, 1804, 1807, 1809, 1811, 1813, 1815, 1818, 1820, 1822, 1851, 1888, 1892, 1894, 1895, 1898, 1899, 1909, 1910, 1911, 1912, 1913
 Script-Objekt Date Zeitzone 181, 1804
 Script-Objekt Enumerator 160
 Script-Objekt Error 160
 Script-Objekt Function 56, 88, 89, 102, 105, 160, 161, 194, 1553, 1650
 Script-Objekt Funktion 1567, 1589, 1650, 1739, 1745
 Script-Objekt Math 62, 150, 160, 195, 209, 1595, 1596, 1620, 1651, 1652, 1672, 1694, 1734, 1739, 1745, 1753, 1769, 1775, 1838, 1839, 1855, 1860, 1874, 1905, 1906, 1909
 Script-Objekt Number 57, 160, 199, 208, 1654, 1657, 1660, 1675, 1909, 1910, 1911
 Script-Objekt Object 58, 149, 160
 Script-Objekt Prototyping 160
 Script-Objekt RegExp 160, 1186
 Script-Objekt Speicherfreigabe 143
 Script-Objekt String 57, 160, 213, 220, 1650, 1738, 1744, 1745, 1746, 1753, 1775, 1827, 1831, 1835, 1837, 1838, 1869, 1877, 1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912
 Script-Objekt var 63, 160
 Script-Objekte 160
 Scriptsprache 482, 561, 567, 572, 597, 609, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 775, 1017, 1028, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Script-String Objekt 213
 SCRIPT-Tag nicht vom Browser erkannt 751
 Script-Versionen 223
 scroll 406, 408, 1581
 Scrollaktion 748, 1906, 1907
 Scrollaktion starten marquee Objekt 747, 1906
 Scrollaktionen Anzahl marquee Objekt 744
 Scrollaktionen Stop marquee Objekt 748, 1907

Scrollbalken 397, 440, 481, 560, 567, 589, 597, 609, 614, 627, 662, 681, 686, 691, 696, 703, 708, 712, 717, 721, 725, 730, 743, 749, 753, 759, 765, 774, 815, 825, 1076, 1129, 1187, 1191, 1592, 1946, 1996
 Scrollbalken Bild Farbe 824
 Scrollbalken Elemente Farbe 814, 824, 1942
 Scrollbalken Farbe 824
 Scrollbalken Kante Farbe 824
 Scrollbalken Kanten Farbe 824
 Scrollbalken Pfeil Farbe 824
 Scrollbalken Pfeil und Bild Farbe 824
 Scrollbalken Pfeil und Kanten Farbe 824
 Scrollbalken Rinne Farbe 824
 Scrollbalken Ziehelement Farbe 815, 824, 1944
 Scrollbar 286
 Scrollbar klicken 483, 1079, 1763
 Scrollbar-Anzeige 610, 663, 1684
 Scrollbar-Anzeige Body 590
 scrollbarDown 483, 1080, 1763
 scrollbarHThumb 483, 1080, 1763
 scrollbarLeft 483, 1080, 1763
 scrollbarPageDown 483, 1080, 1763
 scrollbarPageLeft 483, 1080, 1763
 scrollbarPageRight 483, 1080, 1763
 scrollbarPageUp 483, 1080, 1763
 scrollbarRight 483, 1080, 1763
 scrollbars 397, 440
 Scrollbars 397, 400, 401, 439, 440, 1876
 scrollbarUp 483, 1080, 1763
 scrollbarVThumb 483, 1080, 1763
 Scrollbereich horizontal 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 697, 704, 709, 714, 718, 722, 726, 731, 745, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 Scrollbereich vertikal 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1684
 Scrolleiste 981
 Scrolleisten 400, 401, 1876
 Scrollelemente 483, 591, 611, 777, 1079, 1763
 Scrollelemente Objekt 812, 822, 1937
 scrollen 300, 1876
 scrollen Dokument im Fenster 400, 401, 1876
 scrollen Element 1103
 scrollen endlos marquee Objekt 1653
 scrollen endlos Objekt marquee 1653
 scrollen Hintergrundbild 589, 1581
 scrollen HTML-Element 1103
 scrollen Objekt 812, 822, 1937
 Scrollenbar erzeugen 598, 750, 1684
 scrollender Text 428, 1714
 Scroller in der Statuszeile 287, 385, 1700
 Scrollgeschwindigkeit marquee Objekt 744, 1684
 Scrolling 985
 scrollLeft 742
 Scrollpfeile 1129, 1996
 Scrollrichtung beim Start marquee Objekt 743, 1610
 Scrollrichtung Objekt marquee 1610
 Scrollschrittweite marquee Objekt 744, 1684
 Scroll-Takteinheit marquee Objekt 745, 1721
 Scrolltext 742
 Scrolltext horizontal 742
 Scrolltext vertikal 742



scrollTo() 1876
 Scrollverhalten marquee Objekt 743
 search 735
 Search-Fenster 396, 438
 secue 489
 SECURITY 649, 667, 1685
 Security Zone 974
 securityPolicy 1168
 seeking 866
 SEGMENTTYPE 909, 915, 922, 927, 946, 971, 1687
 sehr klein 987, 989
 sehr-groß 987, 989
 Seite aktuelle 1165
 Seite mit Frame 361, 1145
 Seite verlassen 1104
 Seite vorhergehend Url 428, 1679
 Seite wechseln 1104
 Seite zuvorliegende 1165
 Seiten nachliegende 1165
 Seitenbox 998, 1553
 Seitendarstellung im Dokument 986
 Seiten-Eigenschaften 982
 Seitenelemente nur beim Druck anzeigen 420, 982, 1679
 Seitenelemente vom Druck ausschließen 420, 982, 1679
 Seitenrand 980
 Seitenrand Dokument 584
 Seitenumbruch 986
 Seitenumbruch nach dem Druck des Objektes 813, 823, 1939
 Seitenumbruch vor dem Druck des Objektes 813, 823, 1939
 Seitenvorschub 56, 58, 113, 114, 1184
 SELECT 456, 757, 763, 769, 1096, 1316
 select objekt 248, 463, 465, 1727
 select Objekt 622
 select objekt Auswahl multiple 759, 761, 1658, 1723
 select Objekt Events 1139, 1960
 select Objekt Filter 554
 select Objekt Index der Option 760, 1687
 select Objekt List-Box 760, 1692
 select objekt multiple Auswahl 759, 761, 1658, 1723
 select Objekt Optionindex 760, 1687
 SelectAll 278, 477
 SELECTED 757, 763
 selection Objekt 457, 458, 770, 771, 1082, 1883
 selection Objekt Events 1139, 1960
 selection Objekt Filter 554
 select-multiple 1723
 select-one 1723
 Selektierbarkeit Body 589, 590, 614, 615, 759, 761
 Selektierbarkeit des Objektes 481, 560, 571, 597, 609, 618, 627, 648, 659, 666, 680, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 730, 743, 774, 1014, 1027, 1032, 1036, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1191, 1568
 Selektierbarkeit eines Objektes 482, 563, 568, 573, 599, 610, 619, 628, 649, 660, 667, 682, 688, 692, 697, 701, 705, 709, 714, 718, 723, 727, 731, 745, 754, 1065, 1078, 1192, 1725
 selektieren Objekt 460, 1082, 1085, 1883
 selektierter Text in Auswahlliste 981
 Selektiertheit des Checkbox-Control 691, 1635
 Selektion 276, 434, 457, 459, 476, 771, 1084, 1113, 1130, 1768, 1978, 1998
 Selektion aktive des Dokumentes 770
 Selektion Control-Elemente 456, 770

Selektion Controlrange 457, 771, 1883
 Selektion eines Textbereiches 457, 771, 1883
 Selektion eines Textranges 457, 771, 1883
 Selektion löschen 770, 1764
 Selektion Markierung 770, 1746
 Selektion Text 277, 476
 Selektion Textbereich 457, 771
 Selektion Textrange 457, 771
 Selektion Type 770, 1723
 Selektion und Dokument 458, 770, 1083, 1723, 1724, 1746, 1757, 1764
 Selektion und Option 763
 Selektion Universal-Bereich erzeugen 458, 770, 1083, 1757
 Selektion von Control-Elementen 457, 771, 1883
 Selektionsbereich 591, 1754
 Selektionsfähigkeit eines Objektes 482, 563, 568, 573, 599, 610, 619, 628, 649, 660, 667, 682, 688, 692, 697, 701, 705, 709, 714, 718, 723, 727, 731, 745, 754, 1065, 1078, 1192, 1725
 Selektionsstatus Checkbox-Control 691, 712, 713, 1591, 1606
 Selektionsstatus Default document.select.option Objekt 765, 1607
 Selektionsstatus Default Objekt document.select.option 765, 1607
 Selektionsstatus document.select.option Objekt 766, 1687
 Selektionsstatus eines Control-Elementes 692, 714, 1078, 1699
 Selektionsstatus Objekt document.select.option 766, 1687
 Selektionsstatus Option 766, 1687
 Selektionsstatus Radio Button-Control 691, 712, 713, 1591, 1606
 self 282, 384, 1687
 Semikolon 984
 Senden des Formulars 626
 Senden eines Formulars 702
 Senden/Empfangen der Daten an/von den Server 627, 1656
 Sperator Cookie 177, 425, 489, 1598, 1782, 1784, 1789, 1791, 1793, 1796, 1798, 1800, 1803, 1805, 1807, 1809, 1811, 1814, 1816, 1818, 1820, 1822
 seq 1713
 SEQ 869
 sequentiell 868
 se-resize 988
 SE-resize 993
 Server 49, 695, 699
 Server Datenübersendung 622
 Server Eingabedaten 626, 1556
 Server HTML-Dateien schützen 48
 Server Javascript-Dateien schützen 48
 Server Schutz von HTML-Dateien 48
 Server Schutz von Javascript-Dateien 48
 Server und Cache 1166
 Server Url des Dokumentes 626, 1558
 Servergruppzugehörigkeit Cookie 488
 setCapture() 484, 564, 569, 575, 592, 600, 612, 617, 621, 630, 684, 689, 694, 699, 702, 706, 711, 716, 720, 724, 728, 733, 741, 747, 755, 762, 768, 778, 1027, 1031, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1107, 1147, 1194, 1865
 setExpression() 261, 484, 564, 570, 575, 592, 600, 612, 617, 621, 630, 669, 684, 690, 694, 699, 702, 706, 711,



716, 720, 724, 729, 733, 747, 756, 763, 768, 778, 828, 850, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1061, 1067, 1073, 1081, 1194, 1868
 setInterval() 405
 setzen Focus 591, 616, 761
 setzt Cursor auf Checkbox 634
 setzt Cursor auf Eingabefeld 635
 setzt den Cursor auf das Eingabefeld 634, 641
 setzt den Cursor auf das Radiobutton 637
 setzt den Cursor auf das Resetbutton 637
 setzt den Cursor auf das Submitbutton 638
 Shadow Filter 541
 shadow() 542
 SHAPE 571, 739, 1598
 Shift 1105
 SHIFT_MASK 358, 363, 364, 1105, 1156
 shift-Taste 1154
 Shift-Taste 1154
 SHIFT-Taste links Status 1107, 1689
 SHIFT-Tasten-Status 1107, 1688
 Shockwave 368, 1168
 Shortcuts 1445
 show 327, 329
 showModelessDialog() 379, 381, 1609, 1610
 shuffle 1481
 sich aufblasendes Fenster bei Auflösung von 640x480 299, 399, 1872
 sich selbst schliessen nach Wartezeit 292, 392, 1750
 sichere Verbindung 489
 Sicherheitseinstellungen 1162, 1165, 1901
 Sicherheitseinstellungen im Browser 1162
 Sicherheitseinstellungen vom Netscape 368, 1168
 Sichtbarkeit 985
 Sichtbarkeit des Layers 329
 Sichtbarkeit im Dokument 222
 Sichtbarkeit Objekt 810, 816, 820, 825, 1930, 1948
 Sichtbarkeit Objekt mit Platz im Layout der Umgebung 810, 820, 1930
 Sichtbarkeit Objekt ohne Platz im Layout der Umgebung 816, 825, 1948
 signiertes Script 101, 283, 361, 1146
 Simple Mail Transfer Protocol 563, 737, 754, 1721
 Sinus 199, 1739, 1905
 Site-Enter 509
 Site-Exit 509
 SIZE 725
 slide 1581
 Slide Filter 542
 Slide-Show-Effekt 528, 672
 slideWipe 1704, 1723
 small 987, 989
 smaller 987, 989
 Smart-Tags ab IE 6.x 749
 SMIL 865
 SMTP 563, 737, 754, 1721
 snakeWipe 1704, 1723
 snapshot 864
 sofortiges Abschicken des Formulars 638
 sofortiges Ausführen eines Scriptcodes 211
 Solaris 412, 1672
 solid 983, 987, 991
 Sonderzeichen 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649

Sound 681, 703, 865, 887
 Sound Balance 1574
 Sound im IE 575
 Sound im NS 581
 Sound Wiederholungen 582, 681, 703, 1653
 Sounderzeugung 575
 Soundmixer 885, 908, 914, 921, 927, 932, 945, 954, 971, 1658
 Soundplayer 581
 SourceURL 874, 1469
 Spalte der Tabelle 864
 SPAN 771, 810, 818, 820, 1035, 1319, 1377, 1931
 span Objekt 603, 771
 span Objekt Events 1139, 1960
 span Objekt Filter 554
 span Objekt rendern Internet Explorer 772
 span Objekt rendern Netscape 772
 span Objekt Styles 842
 SPAN rendern Internet Explorer 772
 SPAN rendern Netscape 772
 SPARC 411
 special folder 1445
 Speicherfreigabe Objekt 143
 Speicherfreigabe Script-Objekt 143
 Speichermangel 66, 114
 Speichermangel und Scriptfehler 66
 speichern Dokument 278, 477
 speichern Dokument auf Festplatte 864
 speichern Webseite 1129, 1996
 Speicherplatz einer Variablen 65, 143
 Speicherverwaltung 58
 Spezialzeichen in JScript 113
 Spezialzeichen in Microsoft JScript 113
 Spiral Filter 543
 spiralWipe 1704, 1723
 Sprachausgabe im HTML-Dokument 1498
 Sprachcode des Objektes 561, 736, 1634
 Sprache Browser 1583
 Sprache CSS 826, 1723
 Sprache des Betriebssystems 1169, 1583
 Sprache des Betriebssystems laut UserEinstellung 412, 854, 1727
 Sprache des Cascading Style Sheets (CSS) 826, 1723
 Sprache für Anzeige von Sonderzeichen 482, 561, 567, 572, 597, 601, 609, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Sprache für Anzeige von Sonderzeichen etc. Body 589, 614, 759, 766
 Sprache für Script 482, 561, 567, 572, 597, 609, 618, 627, 648, 659, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 739, 744, 753, 775, 1017, 1028, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Sprache für Script Body 590, 614, 759, 766
 Sprache Installation Betriebssystem 767, 776, 1170, 1708
 Sprache regional des Betriebssystems 1170, 1727
 Sprache Script 1187
 Sprache User des Betriebssystems 1170, 1727
 Sprache zum Objekt 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 727, 745, 767, 776, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 954, 959, 972, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708



Spracheinstellungen 1165, 1901
 Sprunganweisung 92
 Sprungmarke 85, 86
 Sprungziel 561, 572, 736, 1167, 1633
 SQRT1_2 199, 1694
 SQRT2 199, 1694
 src 671
 SRC 309, 325, 451, 613, 670, 671, 756
 src= 46
 s-resize 988
 S-resize 993
 Stamm 222
 Standardbehandlung Ereignis 433, 483, 507, 563, 569,
 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664,
 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723,
 728, 732, 737, 740, 746, 755, 761, 767, 777, 1019,
 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072,
 1079, 1107, 1189, 1193, 1759
 Standardbehandlung für Events 300
 Standard-Behavior 479, 809, 819, 1919
 Standard-Behavior des Internet Explorer 850
 Standard-Behavior und Event 1147
 Standard-Behavior Url im Objekt Style 337, 781
 Standard-Cursor 1617
 Standard-Cursorform im Browser implementiert 337,
 781, 828, 1912, 1930, 1952
 Standard-Dragging-Eigenschaft 1620
 Standard-Druckfenster 398, 1859
 Standard-Eigenschaften und -Methoden aller Objekte in
 JScript 149
 Standard-Eventhandler dem Objekt window zuordnen
 1145
 Standardhierarchie der Events 294
 Standard-IE-Eigenschaften 249, 482, 563, 568, 573, 591,
 599, 602, 610, 616, 619, 628, 649, 660, 663, 667, 682,
 688, 693, 697, 701, 705, 709, 714, 718, 723, 727, 731,
 737, 740, 745, 750, 751, 754, 761, 767, 776, 1019,
 1029, 1033, 1037, 1043, 1048, 1053, 1060, 1066, 1072,
 1079, 1189, 1192, 1196, 1737
 Standarddimension 993
 Standardmethoden aller Objekte 143
 Standard-Methoden aller Objekte in JScript 149
 Standard-Methoden aller Objekte in Microsoft JScript
 149
 Standard-Prioritäten der Operatoren 82
 Standard-Record-Set in einem Datasource-Objekt (DSO)
 753, 1195, 1679
 Standard-Seitenrand Dokument 584
 Standardsprache des Betriebssystems 412, 854, 1708
 Standard-Sprache des Betriebssystems 767, 776, 1170,
 1708
 Standardtext der Statuszeile 378, 1607
 Standardwert laut Objektinitialisierung 686, 691, 696,
 700, 703, 708, 713, 717, 721, 725, 1607
 Standardzeichensatz regionaler 425, 1606
 Start des Download 856, 1906
 Start des Filters 518, 2008
 Start einer VRML-Datei 682, 686, 704, 1696
 Start eines Videoclips 682, 686, 704, 1696
 Start eines Videos 1256
 Start Scrollaktion marquee Objekt 747, 1906
 Startpunkt der Timeline 1714
 Startseite 749
 Startzeit als Offset von der Startzeit der Eltern-Timeline
 1669
 Startzeit der Eltern 1669

starWipe 1703, 1723
 static 984
 status 287, 384, 397, 406, 409, 440, 1700
 Status Body 590, 615, 760, 766
 Status Cookies 1170, 1598
 Status der Filteranimation 516, 2006
 Status der Netzwerkverbindung 411, 1665
 Status der Parsens von Script 1187, 1607
 Status der Timeline 890, 1696, 1698
 Status des Autovervollständigung zum Formular 627,
 708, 725
 Status des Objektes 239, 246, 428, 482, 507, 562, 568,
 572, 583, 590, 598, 602, 615, 619, 628, 649, 660, 663,
 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722,
 726, 731, 736, 740, 744, 750, 751, 753, 760, 766, 775,
 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065,
 1071, 1078, 1188, 1192, 1195, 1679
 Status Drag-Manipulation 563, 574, 591, 599, 602, 611,
 620, 629, 650, 664, 668, 683, 689, 693, 698, 705, 710,
 715, 719, 723, 728, 732, 740, 746, 750, 755, 761, 767,
 777, 1019, 1030, 1034, 1060, 1066, 1080, 1189, 1763
 Status Objekt 1126, 1993
 Statuszeile 378, 384, 397, 406, 409, 440, 1607, 1699
 Statuszeile Hyperlink-Text anzeigen 289, 386, 1702
 Statuszeile im Browserfenster 1637
 Status-Zeile im Browserfenster 1637
 Statuszeile Scroller 287, 385, 1700
 Statuszeile wechselnder Text 287, 385, 1700
 Statuszeile Text Buchstabe für Buchstabe anzeigen 286,
 384, 1699
 Steuertaste und andere Taste 1100
 Steuertasten 1105
 Steuertasten Bitmaske für Zustand 1156
 Steuerungselemente 397, 439
 Stichwortliste 748
 Stichwortliste mehrsprachig 748
 Stop aller Scrollaktionen marquee Objekt 748, 1907
 Stop der Filteranimation 518, 2008
 STOP-Button 1130, 1998
 Stop-Button des Browsers 1104
 stoppen aller Scrollaktionen marquee Objekt 748, 1907
 streaming media file 887, 911, 917, 919, 924, 930, 934,
 948, 957, 973, 1120, 1121, 1987, 1988
 strength 542
 Stretch Filter 544
 Strg 1105
 Strg-Taste 1104, 1153
 Strg-Taste links 1104
 string 54, 77, 81, 143
 String 1376
 String anhängen 281, 602, 1739
 string Basis-Datenstruktur 57
 string Datentyp 57
 String ersetzen 218, 1869
 String JScript-Objekt 149
 String nach Grossbuchstaben konvertieren 220, 1911,
 1912
 String nach Kleinbuchstaben konvertieren 220, 1910,
 1911
 String Objekt 149, 160, 815, 825, 1946
 String oder ein Literal im Unicode-Format dekodieren
 213
 String oder ein Literal in das Unicode-Format kodieren
 144, 153, 211, 1766
 String Script-Objekt 57, 160, 220, 1650, 1738, 1744,
 1745, 1746, 1753, 1775, 1827, 1831, 1835, 1837, 1838,



1869, 1877, 1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912
 String suchen 218, 1838, 1877
 String suchen und ersetzen 218, 1869
 String teilen 219, 1905, 1906, 1908
 String-Literal 213
 String-Literal ersetzen 218, 1869
 String-Literal erzeugen aus Folge von Unicoden 217, 1775
 String-Literal nach Grossbuchstaben konvertieren 220, 1911, 1912
 String-Literal nach Kleinbuchstaben konvertieren 220, 1910, 1911
 String-Literal suchen 218, 1838, 1877
 String-Literal suchen und ersetzen 218, 1869
 String-Literal teilen 219, 1905, 1906, 1908
 String-Literale vergleichen 220, 1838
 String-Literale verketteten 216, 1753
 String-Objekte verketteten 216, 1753
 Stringoperationen 214
 Stringoperator 58
 Strings vergleichen 220, 1838
 String-Variable 213
 Stringverkettung 216, 1753
 Strips Filter 514, 545, 2003
 Stummschaltung Audio 889, 1641
 style 1712
 STYLE 257, 261, 277, 476, 482, 562, 568, 573, 590, 598, 605, 610, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 731, 740, 745, 754, 760, 766, 774, 775, 807, 810, 817, 820, 849, 851, 978, 999, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1100, 1192, 1323, 1703, 1788, 1868, 1929
 Style @font-face 981, 982
 Style @import 982
 Style @import 982
 Style @media 982
 Style @page 818, 1553, 1917
 Style @page Regel 998
 Style 3D-Element threeddarkshadow 981
 Style 3D-Element threedface 981
 Style 3D-Element threedhighlight 981
 Style 3D-Element threedlightshadow 981
 Style 3D-Element threedshadow 981
 Style a:active 984
 Style a:hover 984
 Style a:link 984
 Style a:visited 984
 Style Absatz 984, 985
 Style Auswahlliste highlight 981
 Style Auswahlliste highlighttext 981
 Style background 990
 Style background-attachment 983, 990
 Style background-color 983, 990
 Style background-image 990
 Style background-image: url 983
 Style background-position 983
 Style background-repeat 983, 990
 Style Behavior 479
 Style Behavior Customer-Tag 479
 Style Bildausschnitt 993
 Style Bildschirm 981
 Style border 990
 Style border:eigenschaften_liste 983
 Style border-bottom 983, 990

Style border-bottom-color 991
 Style border-bottom-style 991
 Style border-bottom-width 983, 991
 Style border-color 991
 Style border-left 983, 991
 Style border-left-color 991
 Style border-left-style 991
 Style border-left-width 983, 991
 Style border-right 983, 991
 Style border-right-color 991
 Style border-right-style 991
 Style border-right-width 983, 991
 Style border-style 983, 991
 Style border-top 983, 990
 Style border-top-color 991
 Style border-top-width 983, 991
 Style border-width 983, 991
 Style bottom 984, 992
 Style Bullet 985
 Style caption 989
 Style caption-side 986
 Style Checkbox-Control 690
 Style CLASS-Attribut und ID-Attribut-Wert 805
 Style clear 985
 Style clip 985, 993
 Style color 987, 990, 991, 992
 Style column-gap 987
 Style column-rule 987
 Style column-rule-color 987
 Style column-rule-style 987
 Style column-rule-width 987
 Style column-span 986
 Style colums 987
 Style css 982
 Style cursor 988
 Style Cursor 993
 Style custom Objekt 479
 Style Customer-Tag 479
 Style der ersten Zeile 809, 818, 1917
 Style des ersten Buchstaben 809, 818, 1917
 Style des IE und nicht NS 801
 Style des NS und nicht IE 801
 Style Desktop background 981
 Style direction 985
 Style display 985, 992
 Style Dokumentfenster activeborder 981
 Style Dokumentfenster activecaption 981
 Style Dokumentfenster appworkspace 981
 Style Dokumentfenster buttonface 981
 Style Dokumentfenster buttonhighlight 981
 Style Dokumentfenster buttontext 981
 Style Dokumentfenster captiontext 981
 Style Dokumentfenster greyttext 981
 Style Dokumentfenster inactiveborder 981
 Style Dokumentfenster inactivecaption 981
 Style Dokumentfenster window 981
 Style Dokumentfenster windowframe 981
 Style Dokumentfenster windowtext 981
 Style Druckeigenschaften 993
 Style Drucker 982
 Style dynamischen Eigenschaftenveränderung zur Laufzeit 807
 Style erste Seite 983, 998, 1553, 1678
 Style Farben- und Hintergrundeigenschaften 990
 Style first 983, 998, 1553, 1678
 Style float 797, 810, 820, 849, 985, 1631, 1931



Style font 988, 989
Style font-family 981, 987, 989
Style font-size 986, 987, 989
Style font-style 987, 989
Style font-variant 987, 989
Style font-weight 987, 989
Style footer 983, 998, 1553, 1678
Style format (TrueType) 982
Style Fragezeichen als Joker 988
Style frei definiert als Tag 479
Style Fußnote 983, 998, 1553, 1678
Style global 479
Style h1:first-letter 984
Style h1:first-line 984
Style header 983, 998, 1553, 1678
Style height 992
Style Hintergrundbild fixieren 990
Style Hintergrunddatei 983
Style Hintergrundfarbe 983, 990
Style Hintergrundgrafik 983
Style icon 989
Style input checkbox Objekt 690
Style Internet Explorer IE 6.x und CSS 424, 1595
Style Internet Explorer IE 6.x und CSS1 424, 1595
Style Kodierung von CSS-Werten in Style-Sheets 805
Style Kodierung von Werten mit erlaubter Einheit 805
Style Kommentare innerhalb <STYLE> <STYLE>
804
Style Kopfnote 983, 998, 1553, 1678
Style Layouteigenschaften 990
Style left 983, 984, 992, 998, 1553, 1678
Style letter-spacing 987, 990
Style line-height 986, 990
Style Link 984
Style linke Seite 983, 998, 1553, 1678
Style Liste 985
Style list-style 985
Style list-style-image 985
Style list-style-position 985
Style list-style-type 985
Style local 982
Style margin 986, 991
Style margin-bottom 986, 991
Style margin-left 986, 991
Style margin-right 986, 991
Style margin-top 986, 991
Style Mauscursor 988
Style max-height 985
Style max-width 985
Style menu 989
Style Menü menu 981
Style Menü menutext 981
Style message-box 989
Style min-height 985
Style min-width 985
Style MozBinding (nur NS) 801
Style -moz-opacity 983
Style MozOpacity (nur NS) 801
style Objekt 424, 479, 778, 826, 847, 849, 994, 1595,
1631, 1723
Style Objekt Cursorform 337, 781, 828, 1912, 1930,
1952
Style Objekt Cursorform als Datei 337, 781, 828, 1912,
1930, 1952
Style Objekt custom 479
style Objekt Events 1140, 1961

style Objekt Filter 554
Style Objekt Filter-Attribut 510
style Objekt Styles 842
Style Objekt Url eines Behaviors 337, 781
Style Objekt Wertkonvertierung einer Url 337, 781, 828,
1912, 1951
Style offsetLeft 992
Style offsetTop 992
Style orphans 987
Style overflow 985
Style overflow-x 822
Style p:after 984
Style p:before 984
Style p:first-letter 984
Style p:first-line 984
Style padding 984, 991
Style padding-bottom 984, 991
Style padding-left 984, 991
Style padding-right 991
Style padding-top 984, 991
Style page-break_after 986
Style page-break_before 986
Style page-break-after 993
Style page-break-before 993
Style parentRule (nur NS) 801
Style pixel-left 992
Style pixel-top 992
Style position 984
Style Positionierungsangaben 992
Style print 982
Style Pseudoklasse
first 818, 998, 1553, 1917
left 818, 998, 1553, 1917
right 818, 998, 1553, 1917
Style Pseudoklasse einer @page Regel 998, 1678
Style Pseudoklasse einer Seite 998, 1678
Style Rahmen 983
Style rechte Seite 983, 998, 1553, 1678
Style rect 985
Style RGB-Farbangaben 805
Style right 983, 984, 992, 998, 1553, 1678
Style row-span 986
Style ruby-align 814, 1942
Style ruby-overhang 814, 1942
Style ruby-position 814, 1942
Style Schriftart 987, 989
Style screen 982
Style scrollbar3d-light-color 991
Style scrollbar-arrow-color 991
Style scrollbar-base-color 991
Style scrollbar-dark-shadow-color 991
Style scrollbar-face-color 991
Style scrollbar-highlight-color 992
Style scrollbar-shadow-color 992
Style Scroll-Leiste scrollbar 981
Style Seitenbox 998, 1553
Style Seiten-Selektor 998, 1687
Style Seitenumbruch 986, 987
Style size 986
Style small-caption 989
Style src: url 981, 982
Style status-bar 989
Style StyleSheet-Datei *.css 981
Style StyleSheet-Datei Microsoft 981
Style StyleSheet-Datei Netscape 981
Style Text 987, 990



Style text-align 986, 990
 Style text-decoration 987, 990
 Style Textfarbe 990
 Style text-indent 986
 Style text-shadow 987
 Style Textspalten 987
 Style text-transform 825, 987, 990
 Style Textzeile 986
 Style Tooltip infobackground 981
 Style Tooltip infotext 981
 Style top 984, 992
 Style Trennstrich 987
 Style Überschrift 984, 986
 Style unicode 988
 Style vertical-align 986
 Style Verwendung von " 805
 Style visibility 985, 992
 Style vordefinierte Schriftarten 980
 Style vordefinierte Schriftarten cursive 980
 Style vordefinierte Schriftarten fantasy 980
 Style vordefinierte Schriftarten monospace 980
 Style vordefinierte Schriftarten san-serif 980
 Style vordefinierte Schriftarten serif 980
 Style vordefinierte Style 980
 Style vordefinierte Style italic 980
 Style vordefinierte Style normal 980
 Style vordefinierte Style oblique 980
 Style white-space 987, 990
 Style widow (nicht window !!) 987
 Style width 992
 Style word-spacing 987
 Style word-wrap 990
 Style Wortumbruch 990
 Style Zeichensatz- und Texteingenschaften 989
 Style Zeilenumbruch 990
 Style z-index 992
 Style zoom 353, 796, 992
 style.overflowX 812, 822, 1937
 style.saveSnapshot Behavior 864
 style.textAlign 815, 825, 1946
 style.time2 Behavior 451, 593, 604, 773, 887
 style.time2.par Behavior-Objekt 930
 style.time2.seq Behavior-Objekt 952
 style.userData Behavior 976, 1624, 1837, 1874
 style.visibility 528, 672
 Style-Attribute-Werte 464
 Style-Attribut-Wert 1601
 STYLE-Attribut-Wert im Tag des HTML-Elementes 810, 817, 820, 1929
 Style-Datei 996, 1634
 STYLE-Deklarationen 277, 476
 Style-Dimension % 980
 Style-Dimension cm 980
 Style-Dimension em 980
 Style-Dimension ex 980
 Style-Dimension in 980
 Style-Dimension Inch 980
 Style-Dimension mm 980
 Style-Dimension pc 980
 Style-Dimension Pica 980
 Style-Dimension Point 980
 Style-Dimension pt 980
 Style-Dimension px 980
 Style-Eigenschaft 797

Style-Eigenschaft Wert 253, 257, 261, 264, 483, 484, 564, 565, 570, 574, 575, 592, 593, 600, 601, 611, 612, 613, 617, 618, 620, 621, 622, 629, 630, 631, 668, 669, 683, 684, 689, 690, 693, 694, 695, 698, 699, 701, 702, 706, 707, 710, 711, 712, 715, 716, 719, 720, 721, 724, 725, 728, 729, 733, 734, 746, 747, 755, 756, 762, 763, 768, 769, 777, 778, 826, 827, 828, 849, 850, 1021, 1027, 1030, 1031, 1032, 1034, 1035, 1038, 1039, 1044, 1045, 1049, 1050, 1051, 1054, 1055, 1056, 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1193, 1194, 1195, 1770, 1787, 1867, 1888
 Style-Eigenschaft-Wert 240, 241
 Style-Layout 797
 Style-Layout eines Objektes 849, 1631
 style-Objekt 735
 Style-Regel 994, 1601
 Styles 238
 Styles auf einen Schlag ändern 994
 Styles table.caption Objekt 843
 Styles table.col Objekt 843
 Styles table.colGroup Objekt 844
 Styles table.tBody Objekt 844
 Styles table.tFoot Objekt 844
 Styles table.tHead Objekt 845
 Styles table.tr Objekt 845
 Styles table.tr.td Objekt 845
 Styles table.tr.th Objekt 846
 stylesheet 736, 994, 995, 997
 styleSheet 1331
 Style-Sheet (CSS) 976
 StyleSheet aus externer CSS-Datei importieren 996, 1738
 Style-Sheet Beispiele 979
 Style-Sheet Eigenschaften nur innerhalb <HEAD> deklarieren 977
 Style-Sheet Eigenschaften ohne Tag-Attribut ID deklarieren 978
 StyleSheet entfernen 997, 1868
 StyleSheet importieren 996, 1738
 StyleSheet Mimetyp 996, 1724
 styleSheet Objekt 994, 998, 999, 1553, 1634, 1667, 1668, 1679, 1718, 1724, 1738, 1868
 styleSheet Objekt im Dokument erzeugen 433
 styleSheet Objekt Styles 842
 Style-Sheet und Abstand von Elementen 984
 Style-Sheet und Auswertung von Pixelpositions-Angaben 984
 Style-Sheet- und Dezimal komma 980
 Style-Sheet und Element-Ausschnitt 985
 Style-Sheet und Element-Dimension (-Grenzen, -Anzeigebereich) 984
 Style-Sheet und Elemente-Anordnung im HTML-Dokument 985
 Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet) 985
 Style-Sheet und Element-Scrolling 985
 Style-Sheet und Element-Sichtbarkeit 985
 Style-Sheet und Farbe aktives Fenster 981
 Style-Sheet und Farbe der Scrolleiste 981
 Style-Sheet und Farbe des Desktop 981
 Style-Sheet und Farbe des Dialogfensters 981
 Style-Sheet und Farbe des Dokumentfensters 981
 Style-Sheet und Farbe einer Auswahlliste 981
 Style-Sheet und Farbe eines 3D-Elements 981
 Style-Sheet und Farbe eines Menü 981
 Style-Sheet und Farbe inaktives Fenster 981



Style-Sheet und Farbe von Tooltip und Popuphilfe 981
 Style-Sheet und Font-Dateien (@font-face) 982
 Style-Sheet und Hintergrund-Grafik 983
 Style-Sheet und HTML-Tag-bezogene Eigenschaften 984
 Style-Sheet und Layer-Element 985
 Style-Sheet und numerische (nicht vordefinierte) Farbangaben 980
 Style-Sheet und Rahmen-Eigenschaften (Border) 983
 Style-Sheet und Seiten-Eigenschaften (@page) 982
 Style-Sheet und vordefinierte Bezeichner 980
 Style-Sheet und vordefinierte Dimensionen 980
 Style-Sheet und vordefinierte Farbbereiche 981
 Style-Sheet und vordefinierte Schriften (Font und Style) 980
 Style-Sheet und Wertzuweisung an Eigenschaften 984
 Style-Sheet- Unterklasse und Eigenschaften deklarieren 978
 styleSheet.imports Collection 997, 1738
 styleSheet.pages Collection 998, 1738
 styleSheet.rules Collection 998, 1738, 1868
 Style-Sheet-Beispiele 988
 Style-Sheet-Dateien 981
 Style-Sheet-Deklarations-Varianten 977
 Style-Sheet-Eigenschaften (Style-Sheet-Formatangaben) 984
 Style-Sheet-Eigenschaften für Font 987
 Style-Sheet-Eigenschaften für HTML-Elemente 984
 Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung) 985
 Style-Sheet-Eigenschaften für Mauscursor 988
 Style-Sheet-Eigenschaften für Seitendarstellung im HTML-Dokument 986
 Style-Sheet-Eigenschaften für Tabelle 985
 Style-Sheet-Eigenschaften für Text 986
 Style-Sheet-Eigenschaften für Unicode (Zeichensatz) 988
 Style-Sheet-Eigenschaften mit Attribut STYLE deklarieren 978
 Style-Sheet-Eigenschaften von HTML-Element deklarieren 979
 Style-Sheet-Eigenschaften zu <A> 984
 Style-Sheet-Eigenschaften zu <H1> bis <H6> 984
 Style-Sheet-Eigenschaften zu <P> 984
 Style-Sheet-Format und Style-Sheet-Eigenschaften als tag-unabhängig deklarieren 977
 Style-Sheet-Formatangaben 984
 Style-Sheet-Informationen aus Datei einbinden 981
 Style-Sheet-Objekt erzeugen 241
 Style-Sheet-Objekt erzeugen per Script 239
 Style-Sheet-Objekt im Dokument 253, 826, 1757
 Style-Sheet-Schlüsselwort und Style-Sheet-Eigenschaften als tag-abhängig deklarieren 977
 Style-Sheet-Schriftdatei laden (*.eot bzw. *.prf) 981
 Style-Tag 479
 Style-Tag frei definiert 479
 sub 986
 submit 638, 685, 1721, 1722
 Submit 1097
 SUBMIT 1096
 submit Formular 631, 1908
 Submit Query 638, 721
 Submit wird ausgeführt 1097
 Submit wird nicht ausgeführt 1097
 Submit-Button 1104, 1721
 Submit-Button für Formular 721

substring() 1908
 Suche 1185
 Suche auf Webseiten 562, 572, 1167, 1685
 Suche in einem String 218, 1838, 1877
 Suche in einem String-Literal 218, 1838, 1877
 Suche in einer Webseite 1164, 1844
 Suche in Webseiten 1162
 suchen Text im Textbereich 459, 1084, 1770
 Suchen und Finden 1164, 1844
 Suchfenster 1709
 Such-Fenster 396, 438
 Suchmaschine Steuerung 562, 736, 1679
 Suchmaschinen 49
 Suchmaschinen Scan-Standard 49
 Suchmaschinen zulassen bzw. sperren 49
 Suchmaschinen, die scannen dürfen 49
 Suchmuster 1183, 1186
 Sun 411
 Sun SPARC 1601
 SunOS 412, 1672
 super 986
 Surfen Verlauf 1165
 Surfgewohnheiten des Users als Cookie 485
 switch Anweisung 85, 96
 sw-resize 988
 SW-resize 993
 Synchronisation mit der Timeline 1642
 Synchronisierung von Timelines 878, 879
 Synchronized Accessible Media Interchange 1484
 Synchronized Multimedia Integration Language 865
 Syntax Fehler JScript 115
 Syntax Fehler Microsoft JScript 115
 SyntaxError 194, 1659
 Syntaxfehler 129, 1100, 1147
 System-Clipboard 413, 1108
 SYSTEMLANGUAGE 610, 958
 Systemlautstärke 885, 908, 914, 921, 927, 932, 945, 954, 971, 1658
 System-lokale Einstellungen 158, 212, 1910
 systemnahes Attribut 957
 System-Online-Status 411, 1665
 Systemordner 1433, 1802
 System-Zwischenablage 413
 szeichenfolge 1184
 Szeichenfolge 1184
 t:ANIMATE 890
 t:ANIMATECOLOR 896
 t:ANIMATEMOTION 900
 t:ANIMATION 907
 t:AUDIO 907, 911
 t:EXCL 918, 940
 t:IMG 907, 920
 t:MEDIA 907, 924
 t:PAR 930
 t:PRIORITYCLASS 941
 t:REF 944
 t:SEQ 952
 t:SET 949
 t:SWITCH 958
 t:TRANSITIONFILTER 959
 t:VIDEO 907, 967
 Tabelle 985
 Tabelle Abstand zwischen den Zellen 1014, 1591
 Tabelle Abstand zwischen Zellrahmen und dem Inhalt der Zelle 1014, 1590



Tabelle Änderung der Anzahl Zeilen bzw. Spalten 1026, 1865
 Tabelle Anzahl der Sätze pro Dataset-Anzeige 1014, 1604
 Tabelle Anzahl der sichtbaren Datensätze 1014, 1604
 Tabelle Anzahl der Spalten einer Zelle 1063, 1070, 1595
 Tabelle Anzahl der Spalten in der Tabelle 1014, 1595
 Tabelle Anzahl der Zeilen einer Zelle 1065, 1071, 1683
 Tabelle Anzeige der Tabelle neu erzeugen 1026, 1865
 Tabelle äußerer Rahmen 1017, 1683
 Tabelle Auto-Layout 815, 824, 1944
 Tabelle Datenbereitstellung 1001
 Tabelle Datenbereitstellung in HTML 1001
 Tabelle Datenbereitstellung in JScript 1001
 Tabelle Datenbereitstellung per Active-X-Control 1001
 Tabelle Dateneinbindung 1001
 Tabelle Dateneinbindung in HTML 1001
 Tabelle Dateneinbindung in JScript 1001
 Tabelle Dateneinbindung per Active-X-Control 1001
 Tabelle Datenerzeugung zur Laufzeit 1006
 Tabelle Datensätze 1014, 1019, 1022, 1023, 1025, 1604, 1773, 1835, 1845, 1856
 Tabelle dynamische Daten 1002
 Tabelle dynamische Struktur 1002
 Tabelle erste Seite des Datasets 1019, 1773
 Tabelle erzeugen 1000
 Tabelle erzeugen in HTML 1000
 Tabelle erzeugen in JScript 1001
 Tabelle Fuss erzeugen 1019, 1758
 Tabelle Fuss löschen 1019, 1043, 1759
 Tabelle Gruppierung von Spalten 1065, 1071, 1683
 Tabelle Gruppierung einer Zelle 1065, 1071, 1683
 Tabelle Gruppierung von Zeilen 1065, 1071, 1683
 Tabelle Gruppierung, in der eine Zelle liegt 1065, 1071, 1683
 Tabelle innere Rahmen 1017, 1683
 Tabelle Kommentar 1018, 1704
 Tabelle Kopf erzeugen 1019, 1758
 Tabelle Kopf löschen 1019, 1043, 1759
 Tabelle Lage der Überschrift 1029, 1727
 Tabelle Lage der Zelle 1065, 1071, 1683
 Tabelle Layoutveränderung per Style 1014
 Tabelle letzte Seite des Datasets 1022, 1835
 Tabelle nächste Seite des Datasets 1023, 1845
 Tabelle Rahmen 1016, 1627
 Tabelle Rahmen auf allen Seiten 1016, 1627
 Tabelle Rahmen außen 1017, 1683
 Tabelle Rahmen innen 1017, 1683
 Tabelle Rahmen links 1016, 1627
 Tabelle Rahmen links und rechts 1016, 1627
 Tabelle Rahmen oben und unten 1016, 1627
 Tabelle Rahmen oberhalb 1016, 1627
 Tabelle Rahmen rechts 1016, 1627
 Tabelle Rahmen unterhalb 1016, 1627
 Tabelle Spalte 864
 Tabelle Spalten gruppieren 1035
 Tabelle Spaltengruppe Anzahl der Spalten 1033, 1037, 1692
 Tabelle Spalten-Gruppierung 1065, 1071, 1683
 Tabelle Spaltenüberschrift 1068
 Tabelle Strukturveränderung zur Laufzeit 1006
 Tabelle Trennlinie 1017, 1683
 Tabelle Überschrift erzeugen 1019, 1754
 Tabelle Überschrift Lage 1029, 1727
 Tabelle Überschrift löschen 1019, 1758
 Tabelle und Daten 1001

Tabelle vorhergehende Seite des Datasets 1025, 1856
 Tabelle Zeile einfügen 1022, 1044, 1049, 1055, 1830
 Tabelle Zeile löschen 1049, 1054, 1759
 Tabelle Zeilen tauschen 1023, 1044, 1050, 1055, 1841
 Tabelle Zeilen-Gruppierung 1065, 1071, 1683
 Tabelle Zelle 864
 Tabelle Zelle in Zeile einfügen 1061, 1830
 Tabelle Zelle in Zeile löschen 1060, 1758
 Tabelle Zelle und Gruppierung 1065, 1071, 1683
 Tabellenfuss 1046
 Tabellenkopf 1051
 Tabellenkörper 1040
 Tabellenlayout 812, 815, 822, 824, 1937, 1944
 Tabellen-Objektmodell 1001
 Tabellenspalte 1032
 Tabellenüberschrift 1027
 Tabellenzeile 1056
 Tabellenzelle 1062, 1068
 TABINDEX 435, 483, 563, 568, 569, 573, 574, 591, 599, 611, 616, 620, 629, 649, 650, 660, 661, 668, 683, 688, 689, 693, 697, 698, 705, 710, 715, 719, 723, 727, 728, 732, 745, 746, 754, 755, 761, 776, 777, 1019, 1021, 1030, 1043, 1048, 1049, 1054, 1060, 1066, 1072, 1079, 1080, 1193, 1745, 1775
 TABLE 1332
 table Objekt 864, 1000, 1019, 1022, 1023, 1026, 1043, 1044, 1049, 1050, 1054, 1055, 1590, 1591, 1595, 1604, 1627, 1683, 1704, 1754, 1758, 1759, 1773, 1830, 1835, 1841, 1845, 1856, 1865
 table Objekt Events 1140, 1961
 table Objekt Filter 554
 table Objekt Styles 843
 Table of Contents Identifier einer CD 874, 1469
 table.caption Objekt 1014, 1027, 1590, 1727
 table.caption Objekt Events 1140, 1961
 table.caption Objekt Filter 554
 table.caption Objekt Styles 843
 table.col Objekt 1032, 1692
 table.col Objekt Events 1140, 1961
 table.col Objekt Filter 554
 table.col Objekt Styles 843
 table.colGroup Objekt 1035, 1692
 table.colGroup Objekt Events 1140, 1961
 table.colGroup Objekt Filter 555
 table.colGroup Objekt Styles 844
 table.rows Collection 1039, 1059, 1682
 table.rows.cells Collection 1040, 1063, 1069, 1590
 table.tBodies Collection 1046
 table.tBody Objekt 1019, 1022, 1023, 1040, 1043, 1044, 1049, 1050, 1054, 1055, 1759, 1830, 1841
 table.tBody Objekt Events 1140, 1961
 table.tBody Objekt Filter 555
 table.tBody Objekt Styles 844
 table.tBody.rows Collection 1045, 1059, 1065, 1685
 table.tBody.rows.cells Collection 1045
 table.tFoot Objekt 1018, 1019, 1022, 1023, 1043, 1044, 1046, 1049, 1050, 1054, 1055, 1711, 1759, 1830, 1841
 table.tFoot Objekt Events 1140, 1962
 table.tFoot Objekt Filter 555
 table.tFoot Objekt Styles 844
 table.tFoot.rows Collection 1051, 1059, 1065, 1685
 table.tFoot.rows.cells Collection 1051
 table.tHead Objekt 1018, 1019, 1022, 1023, 1043, 1044, 1049, 1050, 1051, 1054, 1055, 1711, 1759, 1830, 1841
 table.tHead Objekt Events 1141, 1962
 table.tHead Objekt Filter 555



table.tHead Objekt Styles 845
 table.tHead.rows Collection 1056, 1059, 1065, 1685
 table.tHead.rows.cells Collection 1056
 table.tr Objekt 1056, 1682, 1685, 1758, 1830
 table.tr Objekt Events 1141, 1962
 table.tr Objekt Filter 555
 table.tr Objekt Styles 845
 table.tr.td Objekt 1062, 1063, 1065, 1069, 1070, 1071, 1590, 1595, 1683
 table.tr.td Objekt Events 1141, 1962
 table.tr.td Objekt Filter 555
 table.tr.td Objekt Styles 845
 table.tr.th Objekt 1063, 1065, 1068, 1070, 1071, 1590, 1595, 1683
 table.tr.th Objekt Events 1141, 1963
 table.tr.th Objekt Filter 555
 table.tr.th Objekt Styles 846
 Tab-Tasten-Folge 482, 562, 568, 573, 598, 610, 619, 628, 649, 660, 667, 682, 687, 692, 697, 704, 709, 714, 718, 722, 727, 731, 745, 754, 776, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1708
 Tabular Data Container 1001, 1376
 Tabulator 56, 58, 114, 1184
 Tabulator horizontal 56, 58, 113, 114
 Tabulator vertikaler 56, 58, 113, 114
 Tag Behavior 507, 1761
 Tag Customer-Tag 479
 Tag privat 504
 Tag Verhalten 507, 1761
 tag_name.unterklasse_name{eigenschaften_liste} 978
 tag_name:schlüsselwort{eigenschaften_liste} 977
 tag_namen_liste{eigenschaften_liste} 978
 tag-abhängig deklarieren 977
 TAG-Attribute 52
 Tag-Bezeichner 240, 241
 TAG-Bezeichner 52, 244, 561, 567, 572, 590, 598, 601, 614, 619, 627, 649, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 722, 726, 731, 736, 739, 744, 753, 760, 766, 775, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1661
 Tag-Bezeichner Body 590, 615, 761, 767
 Tag-Bezeichner des Objektes 247, 482, 562, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 665, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1708
 Tag-Bezeichner gemeinsam 455, 685, 707, 1040
 Tag-Namen gemeinsam 455, 685, 707, 1040
 Tags mit markierbarem Text 1097
 tag-unabhängig deklarieren 977
 Tangens 199, 1739, 1909
 TARGET 306, 332, 382, 396, 437, 448, 565, 571, 622, 734, 858, 1598, 1659
 Taskleiste von Windows 438
 Tastatur 1106, 1119, 1986
 Tastatur Eventobjekt 1107, 1647
 Tastatur-Ereignisse 1153
 Tastatur-Eventbehandlung 1152
 Tastatur-Eventbehandlung beim IE und Netscape 1156
 Tastatur-Eventbehandlung beim Internet Explorer 1152
 Tastatur-Eventbehandlung beim Netscape 1155
 Tastaturkombinationen 1101
 Tastaturkürzel 729
 Tastaturzugriff auf Body 589, 614, 759

Tastaturzugriff auf ein Objekt per Alt + Taste 481, 560, 567, 571, 589, 597, 609, 614, 618, 680, 686, 691, 695, 703, 707, 712, 717, 721, 725, 730, 743, 752, 759, 774, 1014, 1027, 1032, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1190, 1556
 Taste ASCII-Code 1100, 1105, 1154
 Taste im Dauerdruck 1105
 Taste-Control 686
 Tastenarten 356, 1100, 1101
 Tastendruck 1100
 Tastenkombination 481, 560, 567, 571, 589, 597, 609, 614, 618, 680, 686, 691, 695, 703, 708, 712, 717, 721, 725, 730, 743, 752, 759, 774, 1076, 1190, 1556
 Tastenkombination für Label 1918
 Tastenkombinationen 439
 Tastensimulation 1455
 Tausendertrenner 1376
 TBODY 1040
 TD 742, 811, 821, 822, 1062, 1342, 1935
 TDC 1001, 1376
 teilbar konvertieren 198, 203
 Teilkette 1187
 Teilkette bilden 217, 1827, 1835
 Teilkette einfügen 281, 603, 1830
 Teilkette entfernen 280, 1759
 Teilkette ersetzen 280, 603, 1870
 Teilkette lesen 603, 1909
 Teilmengenprüfung 76
 Teilstring 1185
 Teilzeiger eines Zeigers 48
 Test auf Existenz von parent 383, 1667
 text 639, 685, 988, 993, 1722, 1723
 Text 986, 1108
 TEXT 1096
 Text Abstand zwischen Worten 816, 826, 1949
 Text als Tooltip 571, 680, 686, 703, 1564
 Text auf blinkend prüfen 815, 817, 824, 1945
 Text aus Zwischenablage einfügen 1097, 1103
 Text automatischer Zeilenumbruch 816, 826, 1948
 Text Buchstabe für Buchstabe in Statuszeile anzeigen 286, 384, 1699
 Text dekoratives Layout 815, 824, 1945
 Text des Fenstertitels 382, 1659
 Text durchstreichen 815, 817, 824, 1945
 Text eines Objektes 254, 483, 563, 569, 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664, 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723, 728, 732, 737, 740, 746, 755, 761, 767, 777, 1021, 1030, 1034, 1038, 1043, 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1711, 1776
 Text eines Objektes liefern 242
 Text erste Zeile Style 809, 818, 1917
 Text erster Buchstabe Style 809, 818, 1917
 Text im Dokumentfenster 981
 Text im Textbereich suchen 459, 1084, 1770
 Text markiert ausschneiden und in die Zwischenablage einfügen 1098
 Text markiert in die Zwischenablage kopieren 1097, 1098
 Text markierten ausschneiden und in die Zwischenablage einfügen 1097
 Text Marquee 277, 477
 Text mit eigener Richtung des Renderns 816, 818, 825, 1947
 Text mit HTML-Tags 224
 text Objekt 638



Text ohne HTML-Tags 224, 239, 242
 Text Platz zwischen Worten 816, 826, 1949
 Text positionieren 460, 461, 1085
 Text Position-Ident in der ersten Textzeile 815, 825, 1946
 Text Rahmen um Text 815, 825, 1946
 Text rendern 816, 818, 825, 1947
 Text scrollend 428, 1714
 Text Style der ersten Zeile 809, 818, 1917
 Text Style des ersten Buchstaben 809, 818, 1917
 Text überstreichen 815, 818, 824, 825, 1945
 Text unterstreichen 815, 818, 824, 825, 1945
 Text unterstreichen Position des Striches 815, 825, 1947
 Text von Tooltips und Popuphilfen 981
 Text Wortumbruch bei Überschreitung der Objektgrenzen 816, 826, 1949
 Text Zeilenumbruch automatisch 816, 826, 1948
 Text Zeilenumbruch in Worten 816, 826, 1949
 Text zu einem Hyperlink (HREF) für eine feste Zeispanne in Statuszeile anzeigen 289, 386, 1702
 text/css 736, 994, 995, 996, 997
 text/ecmascript 1723
 text/html 370, 437, 438, 1169, 1171, 1555
 text/javascript 46, 1556, 1723
 text/Jscript 1723
 text/plain 438
 text/tcl 1556
 text/vbs 1723
 text/vbscript 1556, 1723
 text/xml 1723
 text/x-scriptlet 1287
 Textabstand bei asiatischen Zeichen 815, 824, 1945
 Textarea 1097
 TEXTAREA 1074, 1076, 1096, 1108, 1353, 1582, 1595, 1647
 Textarea Anzahl der sichtbaren Zeichen 1076, 1595
 Textarea Anzahl der sichtbaren Zeilen 1078, 1683
 Textarea Mehrzeiligkeit 1078, 1683
 textarea Objekt 457, 458, 460, 622, 770, 771, 827, 1074, 1082, 1084, 1595, 1607, 1683, 1722, 1733, 1853, 1883
 textArea Objekt 1078, 1699
 textarea Objekt Events 1142, 1963
 textarea Objekt Filter 555
 textarea Objekt Styles 846
 Textarea Vorbelegung 1076, 1607
 Textarea Wortumbruch 1078, 1733
 Textausrichtung 815, 824, 980, 1944
 Textausrichtung Blocksatz 815, 825, 1946
 Textausrichtung der letzten Zeile 815, 824, 1945
 Textausschnitt 1096
 Textbereich 458, 591, 599, 688, 701, 710, 719, 723, 728, 1079, 1082, 1757
 Textbereich bewegen 460, 1084, 1842
 Textbereich Bookmark 459, 460, 1084, 1778, 1842
 Textbereich der Selektion 457, 771
 Textbereich des Elternobjektes 240, 245, 482, 562, 568, 572, 590, 598, 602, 615, 619, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 750, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668
 Textbereich des Elternobjektes Body 590, 615, 760, 766
 Textbereich duplizieren 459, 1084, 1764
 Textbereich Elternelement 460, 1084, 1850
 Textbereich Erzeugbarkeit 627, 1647
 Textbereich HMTL-Text im Bereich 459, 1083, 1634

Textbereich im Dokument 458, 1082
 Textbereich Inhalt ersetzen 460, 827, 1084, 1853
 Textbereich Inhalt füllen 460, 827, 1084, 1853
 Textbereich Inhalt leeren 460, 827, 1084, 1853
 Textbereich mit HTML-Code 460, 827, 1084, 1853
 Textbereich Plain-Text dehnen 459, 1084, 1770
 Textbereich Plain-Text im Bereich 459, 1083, 1711
 Textbereich Rechteck um Textbereich 459, 1083, 1582
 Textbereich Rechteckes um Textbereich 459, 1083, 1583
 Textbereich Selektion 457, 771, 1883
 Textbereich Text suchen 459, 1084, 1770
 Textbereich Textmarke 459, 460, 1084, 1778, 1842
 Textbereich Zeiger auf Elternelement 460, 1084, 1850
 Textbereiche 980
 Textbereiche prüfen 460, 1084, 1828, 1830
 Textbereich-Zeichen-Zeiger 459, 1083, 1751
 Textblock 770
 text-bottom 986
 Textbox 709, 727, 1731
 Textbox Editierbarkeit 1679
 Textbox einzeilige 725
 Textbox Länge 708, 726, 1654
 Text-Control 707, 709, 727, 1722, 1731
 Text-Control Editierbarkeit 1679
 Textcontrol einzeilig 725
 Text-Control einzeilig 1722
 Text-Control Länge 708, 726, 1654
 Text-Control maximale Anzahl der durch User eingebbarer Zeichen 708, 726, 1654
 Textdaten 280
 Textdaten anhängen 281
 Textdaten einfügen 281
 Textdaten ersetzen 280
 Textdaten erzeugen 280
 Textdaten löschen 280
 Textdaten-Übergabe durch an Url angehängte Textdaten 317, 323, 646, 657
 Textdaten-Übergabe durch Fenster-Handle 318, 324, 647, 658
 Textdekoration 815, 817, 818, 824, 825, 1945, 1947
 Textdekoration abschalten 815, 817, 825, 1945
 Text-Eingabe-Control 1074
 Text-Eingabe-Control mehrzeilig 1074
 Textelement erzeugen 242
 Textelement erzeugen per Script 239
 Textelemente 1102
 Textfarbe 277, 426, 476, 810, 820, 1625, 1928
 Textfarbe Body 590, 1711
 Textfeld 709, 727, 1731
 Textfeld - eingegebenen Text selektieren 641
 Textfeld Editierbarkeit 1679
 Textfeld für Eingabe aktivieren 641
 Textfeld für Eingabe deaktivieren 641
 Textfeld Länge 708, 726, 1654
 Textfluss für asiatische Zeichen 811, 820, 1933
 Textfont 811, 820, 1931
 Textfont Familie 811, 820, 1932
 Textfont Fettheit 811, 820, 1932
 Textfont Höhe 811, 820, 1932
 Textfont Stil 811, 820, 1932
 Textfont Stil-Variante 811, 820, 1932
 Textknoten 240, 244, 280, 1017, 1028, 1032, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1661
 Textkonvertierung 815, 825, 1946



Textkonvertierung jeder Wortanfang mit Grossbuchstabe 815, 825, 1946
 Textkonvertierung nach Großbuchstaben 815, 825, 1946
 Textkonvertierung nach Kleinbuchstaben 815, 825, 1946
 Textbereich-Zeichen-Zeiger ändern 460, 1084, 1085, 1839, 1842, 1888
 Textmarke im Textbereich 459, 1084, 1778
 Textmarke Textbereich 460, 1084, 1842
 Textmarkierung 1104
 TextNode Objekt 280, 458, 1082
 Text-Objekt im Dokument erzeugen 433
 Textrange 458, 1082
 TextRange 1357
 Textrange der Selektion 457, 771
 textrange Objekt 457, 458, 459, 460, 770, 771, 827, 1083, 1084, 1085, 1582, 1583, 1634, 1711, 1751, 1757, 1764, 1770, 1778, 1828, 1830, 1839, 1842, 1850, 1853, 1883, 1888
 TextRange Objekt 280, 458, 1082
 textrange Objekt Events 1142, 1963
 textrange Objekt Styles 847
 Textrange Selektion 457, 771, 1883
 TextRange.TextRectangle Collection 458, 460, 1082, 1085
 TextRange.TextRectangle Objekt 458, 461, 1082, 1085
 TextRectangle 460, 461, 591, 616, 762, 768, 1085
 textrectangle Objekt 1581, 1649, 1682, 1720
 textrectangle Objekt Styles 847
 TextRectangle-Objekt 460, 483, 564, 569, 574, 600, 602, 611, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1778
 Textselektion 277, 476
 Textspalten 987
 Textstream 1442
 Textsuche im Dokument 1162
 Textsuche in einer Webseite 1164, 1844
 text-top 986
 Textumfluss 810, 815, 817, 818, 819, 820, 824, 1927, 1931, 1944
 Textzeichen Abstand 811, 821, 1934
 Textzeichen-Layout-Gitter 811, 820, 1933
 Textzeichen-Layout-Gitter 2D 811, 821, 1933
 Textzeichen-Layout-Gitter Linie 811, 821, 1933
 Textzeichen-Layout-Gitter Typ 811, 821, 1934
 Textzeichen-Layout-Gitter Zeichengröße 811, 820, 1933
 Textzeile Abstand zweier Objekte oder zweier Textzeilen 811, 821, 1934
 Textzeile positionieren 460, 461, 1085
 TFOOT 1019, 1043, 1046, 1059, 1065, 1685, 1758, 1759
 TH 1068, 1346
 THEAD 1019, 1043, 1051, 1059, 1065, 1685, 1758, 1759
 thick 991
 thin 991
 this 77
 this Anweisung 84, 97, 150, 209
 this Operator 77
 throw 193
 throw Anweisung 97, 98, 129
 Tilde 75, 76
 Time Container 868
 Time Formate 872
 Time Gruppierung von Elementen 865

time2 Behavior 1713
 TIMEACTION 560, 561, 571, 572, 597, 609, 614, 618, 627, 666, 681, 686, 691, 695, 696, 700, 703, 708, 712, 713, 717, 721, 725, 743, 765, 774, 868, 885, 893, 897, 904, 907, 908, 913, 914, 918, 920, 921, 926, 931, 936, 937, 940, 944, 945, 949, 954, 965, 970, 971, 1014, 1016, 1027, 1028, 1041, 1046, 1051, 1052, 1057, 1058, 1063, 1064, 1069, 1070, 1076, 1077, 1191, 1579, 1619, 1621
 timeChildren Collection 865, 890
 Time-Container 865, 866
 Time-Container erzeugen 868
 Timed Interactive Multimedia Extensions 865
 Timeline Clip 908, 913, 921, 926, 944, 970, 1593
 Timeer Synchronisation 866
 Time-Formate Behavior .style.time2 872
 Time-Formate.style.time2 Behavior 872
 Timeline 559, 560, 571, 597, 609, 610, 614, 618, 627, 666, 681, 686, 691, 695, 700, 703, 708, 712, 717, 721, 725, 743, 765, 774, 865, 885, 887, 893, 897, 904, 907, 909, 913, 915, 920, 922, 926, 927, 932, 936, 944, 946, 949, 954, 965, 970, 972, 1014, 1027, 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1131, 1191, 1579, 1642, 1711, 1999
 Timeline Beschleunigung des Elementes auf der Timeline 885, 892, 896, 903, 907, 913, 918, 920, 925, 931, 944, 954, 964, 969, 1555
 Timeline 1D-Animation 893, 897, 904, 1586
 Timeline 2D-Animation 893, 894, 897, 898, 900, 904, 905, 965, 1586, 1647, 1648, 1728
 Timeline 2D-Animation per Vektorgraphik 904, 1670
 Timeline Abbruch bzw. Pausierung der aktiven Animation 943, 1633, 1654, 1671
 Timeline Abbruch der Synchronisation 911, 917, 924, 930, 934, 949, 957, 974, 1131, 1999
 Timeline Aktion 867
 Timeline Aktion eines Elementes zum Zeitpunkt des Ende der Timeline 885, 893, 897, 904, 908, 914, 921, 926, 931, 945, 949, 954, 971, 1626
 Timeline aktiv 878, 1642
 Timeline aktiver Playlisten-Eintrag 934, 937
 Timeline aktiver Track 934, 937
 Timeline aktives Element animieren ab einem Zeitpunkt auf der Segment-Timeline 886, 895, 899, 906, 910, 916, 919, 923, 929, 933, 947, 950, 951, 956, 967, 973, 1878, 1879
 Timeline aktives Element animieren ab einem Zeitpunkt auf der Timeline 886, 895, 899, 906, 910, 916, 923, 929, 933, 947, 950, 956, 966, 973, 1877
 Timeline aktives Element auf der aktiven Timeline beginnt gerade zu pausieren 887, 895, 900, 906, 911, 917, 924, 930, 934, 948, 951, 957, 967, 974, 1125, 1992
 Timeline aktives Element auf der Timeline nach einer Zeitspanne der Aktivität des Elementes ab Beginn der Timeline stoppen 886, 894, 899, 905, 910, 916, 923, 928, 932, 947, 950, 955, 966, 972, 1765
 Timeline aktives Element auf der Timeline stoppen 886, 894, 899, 905, 910, 916, 919, 923, 928, 932, 947, 950, 955, 966, 972, 1764
 Timeline aktives Element auf Timeline pausieren lassen 886, 894, 899, 905, 910, 916, 923, 929, 933, 947, 950, 955, 966, 973, 1853
 Timeline aktives Element pausiert 887, 895, 900, 906, 911, 917, 924, 930, 934, 948, 951, 957, 967, 974, 1125, 1992



Timeline aktivieren 887, 895, 899, 906, 910, 916, 924, 929, 933, 948, 951, 956, 967, 973, 1114, 1979
Timeline Aktivität 889
Timeline aktuelle Breite des Media-Elementes in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
Timeline aktuelle Höhe des Media-Elementes in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
Timeline aktueller prozentualer Status des Pufferns einer Media-Datei 907, 913, 920, 926, 944, 970, 1583
Timeline aktueller Punkt auf Timeline 1691
Timeline aktueller Zeitpunkt 889, 1560
Timeline aktueller Zeitpunkt in der Timeline der Wiederholungen laut autoReverse 1686
Timeline alle Timeline-Einstellungen zum Element aufheben 886, 894, 899, 906, 910, 916, 919, 923, 929, 933, 947, 950, 956, 973, 1871
Timeline Animation einer Media-Datei 924
Timeline Animation eines beliebigen Elementes 907
Timeline Animation eines Elementes anhand einer speziellen Style-Eigenschaft 890
Timeline Animation eines Elementes exklusiv 917
Timeline Animation eines HTML-Elementes 949
Timeline Animation exklusiv 868
Timeline Animation parallel 868, 930, 940
Timeline Animation sequentiell 868, 952
Timeline Animationsart 868
Timeline animieren Bild 920
Timeline animieren Image 920
Timeline Anzahl der bisher downgeloadeten Bytes beim Laden einer Media-Datei 908, 914, 921, 926, 945, 970, 1616
Timeline Anzahl der Bytes einer Media-Datei 908, 914, 921, 926, 945, 970, 1616
Timeline Anzahl der Dauer in Sekunden für Wiederholung auf der Timeline 885, 893, 898, 904, 909, 915, 918, 922, 927, 932, 945, 949, 954, 965, 971, 1680
Timeline Art der Animation 868, 893, 897, 904, 1586
Timeline Art des Updates der Eigenschaften eines Elementes 909, 915, 922, 928, 946, 972, 1725
Timeline Audio aktiv oder aus (stumm) auf der Timeline 885, 908, 914, 921, 927, 932, 945, 954, 971, 1658
Timeline Audio Wiedergabe 911
Timeline Audio-Element Lautstärkeeinstellung 908, 914, 921, 926, 945, 971, 1630
Timeline Audio-Element Stummschaltung 908, 914, 921, 926, 945, 971, 1630
Timeline Audioinhalt einer Media-Datei 908, 914, 921, 926, 945, 971, 1630
Timeline ausblenden eines Elementes 959
Timeline Auswahl eines Zeitpunktes einer Media-Datei 907, 913, 921, 926, 944, 970, 1590
Timeline automatische Rückwärtsanimation eines Elementes 885, 893, 897, 904, 907, 913, 918, 920, 925, 931, 944, 949, 954, 965, 970, 1573
Timeline beenden 887, 895, 899, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1983
Timeline beliebigen Elementes animieren 907
Timeline Beschreibung einer Media-Datei 907, 913, 920, 925, 936, 944, 969, 1553
Timeline Bezeichner einer Style-Eigenschaft eines Elementes 893, 897, 949, 1568
Timeline Bezugspunkt der Animation 904, 1666
Timeline Bild animieren 920
Timeline Bildschirmanzeige Element 908, 914, 921, 927, 945, 971, 1632

Timeline Breite des Media-Elementes in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
Timeline Bytes einer Media-Datei 908, 914, 921, 926, 945, 970, 1616
Timeline Clip 908, 914, 921, 926, 944, 970, 1593
Timeline Collection playList für Element auf der Timeline 914, 927, 945, 971, 1632
Timeline Container 868
Timeline Copyright der Media-Datei 908, 914, 921, 926, 936, 944, 970, 1599
Timeline Datenfluss einer Media-Datei 908, 914, 921, 927, 945, 971, 1646
Timeline Daten-Stream 908, 914, 921, 926, 945, 970, 1617
Timeline Dauer 867, 889, 1559
Timeline Dauer der Timeline der Wiederholungen laut autoReverse 1686
Timeline Dauer einer Wiederholung 1690
Timeline des Dokumentes 866
Timeline Dokument mit Referenz 944
Timeline Download Start einer Media-Datei 908, 914, 921, 926, 945, 971, 1630
Timeline einblenden eines Elementes 959
Timeline Element Abbruch bzw. Pausierung der aktiven Animation 943, 1633, 1654, 1671
Timeline Element aktiv 878
Timeline Element als Referenz 944
Timeline Element anhand einer speziellen Style-Eigenschaft animieren 890
Timeline Element Animation parallel 940
Timeline Element Animation sequentiell 952
Timeline Element animieren 907
Timeline Element animieren exklusiv 917
Timeline Element anzeigen auf Bildschirm 908, 914, 921, 927, 945, 971, 1632
Timeline Element Art des Updates der Eigenschaften 909, 915, 922, 928, 946, 972, 1725
Timeline Element auf der Timeline aktivieren 887, 895, 899, 906, 910, 916, 924, 929, 933, 948, 951, 956, 967, 973, 1114, 1979
Timeline Element auf der Timeline starten, also aktivieren 886, 894, 898, 905, 909, 915, 919, 923, 928, 932, 946, 950, 955, 966, 972, 1742
Timeline Element auf der Timeline zu einem spezifischen Zeitpunkt ab Start der Timeline starten (aktivieren) 886, 894, 898, 905, 910, 916, 923, 928, 932, 946, 950, 955, 966, 972, 1743
Timeline Element ausblenden 959
Timeline Element auswählen automatisch 957
Timeline Element automatisch aktivieren 957
Timeline Element automatisch auswählen 957
Timeline Element beenden auf der Timeline 887, 895, 899, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1983
Timeline Element Bezeichner einer Style-Eigenschaft 893, 897, 949, 1568
Timeline Element Bildschirmanzeige 908, 914, 921, 927, 945, 971, 1632
Timeline Element einblenden 959
Timeline Element Eltern-Timer 870
Timeline Element exklusiv animieren 917
Timeline Element Farbanimation 896
Timeline Element handhand spezieller Style-Eigenschaft animieren 878
Timeline Element in HTML Animation 949
Timeline Element inaktiv 878



Timeline Element initialisieren 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 974, 1127, 1994

Timeline Element Kind-Timer 870

Timeline Element kumulativ animieren 892, 896, 903, 964, 1556

Timeline Element Lautstärkeeinstellung 908, 914, 921, 926, 945, 971, 1630

Timeline Element mit seiner Timeline synchronisieren 911, 917, 924, 930, 934, 949, 957, 974, 1131, 1999

Timeline Element mit systemnaheem Attribut 957

Timeline Element mit Timer versorgen 876

Timeline Element nicht synchron zur seiner Timeline 911, 917, 924, 930, 934, 948, 957, 974, 1124, 1991

Timeline Element parallele Animation 940

Timeline Element Pause aufheben 887, 895, 900, 906, 911, 917, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1995

Timeline Element rückwärts animieren auf der Timeline 866

Timeline Element rückwärts auf der Timeline animieren 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1996

Timeline Element sequentiell animieren 952, 955, 1845, 1855

Timeline Element sichtbar machen 959

Timeline Element stoppen auf der Timeline 887, 895, 900, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1983

Timeline Element Stummschaltung 908, 914, 921, 926, 945, 971, 1630

Timeline Element und Collection playList 914, 927, 945, 971, 1632

Timeline Element unsichtbar machen 959

Timeline Element Updates der Eigenschaften 909, 915, 922, 928, 946, 972, 1725

Timeline Element verliert seine Timeline 911, 917, 924, 930, 934, 948, 957, 974, 1124, 1991

Timeline Element Widergabe per Media Bar-Player 1673

Timeline Element Widergabe per Windows Media Player 1673

Timeline Element Wiederholung der Animation 887, 895, 900, 906, 911, 917, 919, 924, 930, 934, 948, 951, 957, 967, 974, 1126, 1994

Timeline Element zurücksetzen 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 974, 1127, 1994

Timeline Element Zustand 866

Timeline Eltern-Timer 870

Timeline Ende 867

Timeline Ende der aktiven Timeline erreicht 887, 895, 899, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1983

Timeline Ende der Animation von Elementen in einem gemeinsamen Time-Container 881, 918, 931, 1623

Timeline Ende von Elementen in einem gemeinsamen Time-Container 881, 918, 931, 1623

Timeline Endezeit als Offset von der Startzeit der Eltern-Timeline 1669

Timeline Endezeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes 908, 914, 921, 926, 944, 970, 1593

Timeline endezeitpunkt innerhalb des Media-Elementes 908, 914, 921, 926, 944, 970, 1593

Timeline Endezeitpunkt Media-Element 908, 914, 921, 926, 944, 970, 1593

Timeline Endwert zur Werterhöhung 894, 898, 905, 950, 1718

Timeline erzeugen 867

Timeline Event Seek-Methode 887, 911, 917, 920, 924, 930, 934, 949, 957, 967, 974, 1129, 1996

Timeline exklusive Animation eines Elementes 917

Timeline Farbanimation eines Elementes 896

Timeline Farbwerte für die Elemente-Animation 898, 1729

Timeline Filteranimation als "Übergang" 959

Timeline Fortschrittsangabe entlang der Timeline 1676

Timeline Frame eines aktiven Elementes auf der Timeline anwählen 886, 895, 899, 906, 910, 916, 919, 923, 929, 933, 948, 951, 956, 973, 1880

Timeline für ein Objektgruppe erzeugen 868

Timeline für eine Objektgruppe aus einem unverschachtelten Timer 869

Timeline für eine Objektgruppe aus verschachtelten Timern 870

Timeline generelle Möglichkeit der Auswahl eines Zeitpunktes einer Media-Datei 907, 913, 921, 926, 944, 970, 1590

Timeline generelle Pausierungsmöglichkeit einer Media-Datei 907, 913, 920, 926, 944, 970, 1590

Timeline generelle Restartmöglichkeit eines Elementes auf der Timeline 885, 893, 898, 904, 909, 915, 922, 927, 932, 946, 949, 954, 966, 971, 1681

Timeline halten 1642

Timeline Höhe des Media-Elementes in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655

Timeline HTML-Element Animation 949

Timeline ID des zu animierenden Elementes 894, 898, 905, 950, 966, 1709

Timeline Image animieren 920

Timeline Implementierung eines Timers bzw. Neusetzung seiner Attribute 878

Timeline Import des Behavior .style.time2 876

Timeline inaktiv 878

Timeline Index des Objektes playItem in der Collection playList 1635

Timeline Index des Playlisten-Eintrages in der Collection playList 937, 1636

Timeline Inhalt einer Media-Datei 908, 914, 921, 926, 927, 945, 971, 1630, 1632

Timeline initialisieren 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 974, 1127, 1994

Timeline Kind-Timer 870

Timeline Konvertierung 886, 887, 894, 895, 898, 899, 905, 906, 909, 910, 915, 916, 919, 922, 923, 924, 928, 929, 932, 933, 946, 947, 948, 950, 951, 955, 956, 966, 967, 972, 973, 1734, 1736, 1760, 1850, 1851, 1882, 1904

Timeline Konzept 865

Timeline Konzept Alternativen 866

Timeline Länge der Media-Datei in Sekunden 908, 914, 921, 927, 931, 945, 954, 971, 1655

Timeline Lautstärke eine Audio-Elementes auf der Timeline relativ zur Lautstärke des Elternelementes 885, 890, 909, 915, 922, 928, 932, 946, 955, 972, 1732

Timeline Lautstärkeeinstellung 908, 914, 921, 926, 945, 971, 1630

Timeline Media-Datei animieren 924

Timeline Media-Datei Anzahl Bytes 908, 914, 921, 926, 945, 970, 1616



Timeline Media-Datei Auswahl eines Zeitpunktes 907, 913, 921, 926, 944, 970, 1590
 Timeline Media-Datei Beschreibung 907, 913, 920, 925, 936, 944, 969, 1553
 Timeline Media-Datei Bytes 908, 914, 921, 926, 945, 970, 1616
 Timeline Media-Datei Copyright 908, 914, 921, 926, 936, 944, 970, 1599
 Timeline Media-Datei Daten-Stream 908, 914, 921, 926, 945, 970, 1617
 Timeline Media-Datei Download Start 908, 914, 921, 926, 945, 971, 1630
 Timeline Media-Datei Länge in Sekunden 908, 914, 921, 927, 931, 945, 954, 971, 1655
 Timeline Media-Datei MIME-Typ 909, 915, 922, 928, 937, 946, 972, 1723
 Timeline Media-Datei mit Audioinhalt 908, 914, 921, 926, 945, 971, 1630
 Timeline Media-Datei mit Datenfluss 908, 914, 921, 927, 945, 971, 1646
 Timeline Media-Datei mit visuellem Inhalt 908, 914, 921, 927, 945, 971, 1632
 Timeline Media-Datei Name des Autor 907, 913, 920, 925, 936, 944, 969, 1570
 Timeline Media-Datei Pausierungsmöglichkeit 907, 913, 920, 926, 944, 970, 1590
 Timeline Media-Datei puffern 907, 913, 920, 926, 944, 970, 1583
 Timeline Media-Datei Rating 915, 922, 927, 937, 945, 971, 1678
 Timeline Media-Datei Start Download 908, 914, 921, 926, 945, 971, 1630
 Timeline Media-Datei Status Laden 908, 914, 921, 926, 945, 970, 1616
 Timeline Media-Datei Titel 909, 915, 922, 927, 937, 946, 972, 1716
 Timeline Media-Datei Url 909, 915, 922, 927, 937, 946, 971, 1695
 Timeline Media-Element 887, 911, 917, 919, 924, 930, 934, 948, 956, 957, 973, 1120, 1121, 1987, 1988
 Timeline Media-Element aktuelle Breite in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
 Timeline Media-Element aktuelle Höhe in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
 Timeline Media-Element Breite in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
 Timeline Media-Element Endezeitpunkt 908, 914, 921, 926, 944, 970, 1593
 Timeline Media-Element Höhe in Pixels 908, 914, 921, 927, 931, 945, 954, 971, 1655
 Timeline Media-Element MIME-Typ 908, 914, 921, 927, 945, 971, 1656
 Timeline Media-Element Start der Wiedergabe 908, 914, 921, 927, 945, 971, 1649
 Timeline Media-Element Startzeitpunkt 908, 913, 921, 926, 944, 970, 1593
 Timeline Media-Element Wartezeit für Start der Wiedergabe 908, 914, 921, 927, 945, 971, 1649
 Timeline Media-Element Wiedergabeplayer 908, 914, 921, 927, 945, 971, 1672
 Timeline Media-Element Wiedergabeplayer Zeiger 909, 914, 922, 927, 945, 971, 1674
 Timeline Medium zum Element ist komplett geladen 887, 911, 917, 924, 930, 934, 948, 956, 973, 1120, 1987

Timeline Medium zum Element ist wegen Fehler nicht geladen 887, 911, 917, 919, 924, 930, 934, 948, 957, 973, 1121, 1987
 Timeline MIME-Typ der Media-Datei 909, 915, 922, 928, 937, 946, 972, 1723
 Timeline MIME-Typ des Media-Elementes 908, 914, 921, 927, 945, 971, 1656
 Timeline nächstes playItem Objekt aktivieren laut Collection playList 928, 932, 940, 947, 955, 1847
 Timeline Name des Autor der Media-Datei 907, 913, 920, 925, 936, 944, 969, 1570
 Timeline Objektgruppe 868
 Timeline parallele Animation 930, 940
 Timeline Pause aufheben 887, 895, 900, 906, 911, 917, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1995
 Timeline Pause eines aktiven Elementes aufheben 887, 895, 900, 906, 911, 917, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1995
 Timeline Pause eines aktiven Elementes, das auf der Timeline pausiert, aufheben 886, 894, 899, 906, 910, 916, 923, 929, 933, 947, 950, 956, 966, 973, 1873
 Timeline pausieren 592, 1645, 1853
 Timeline pausieren beenden 1873
 Timeline Pausierungsmöglichkeit einer Media-Datei 907, 913, 920, 926, 944, 970, 1590
 Timeline Performance Browser 876
 Timeline Playliste 911
 Timeline Playlisten-Eintrag 934, 937
 Timeline Playlisten-Eintrag aktiv 934, 937
 Timeline Playlisten-Eintrag-Index in der Collection playList 937, 1636
 Timeline Rating der Media-Datei 915, 922, 927, 937, 945, 971, 1678
 Timeline Referenz innerhalb des Dokumentes 944
 Timeline rückwärts 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 967, 974, 1128, 1996
 Timeline Schrittweite der Werterhöhung bzw. Art der Animation 893, 897, 904, 1586
 Timeline Scriptkommando 917, 928, 930, 972, 1131, 2000
 Timeline Seek-Methode zum Element 887, 911, 917, 920, 924, 930, 934, 949, 957, 967, 974, 1129, 1996
 Timeline sequentielle Animation 952
 Timeline sichtbar machen eines Elementes 959
 Timeline spezielle Style-Eigenschaft für Animation 878
 Timeline Start der Wiedergabe Media-Element 908, 914, 921, 927, 945, 971, 1649
 Timeline Start des Downloads einer Media-Datei 908, 914, 921, 926, 945, 971, 1630
 Timeline Startpunkt 1714
 Timeline Startwert zur Werterhöhung 893, 898, 904, 1628
 Timeline Startzeit als Offset von der Startzeit der Eltern-Timeline 1669
 Timeline Startzeitpunkt eines Unterbereiches (Clip) innerhalb des Media-Elementes 908, 913, 921, 926, 944, 970, 1593
 Timeline Startzeitpunkt innerhalb des Media-Elementes 908, 913, 921, 926, 944, 970, 1593
 Timeline Startzeitpunkt Media-Element 908, 913, 921, 926, 944, 970, 1593
 Timeline Status 890, 1696, 1698
 Timeline Status des Pufferns einer Media-Datei 907, 913, 920, 926, 944, 970, 1583
 Timeline Status Laden einer Media-Datei 908, 914, 921, 926, 945, 970, 1616



Timeline stoppen 887, 895, 900, 906, 911, 917, 919, 924, 929, 933, 948, 951, 956, 967, 973, 1117, 1983
 Timeline streaming media file 887, 911, 917, 919, 924, 930, 934, 948, 957, 973, 1120, 1121, 1987, 1988
 Timeline Stummschaltung 908, 914, 921, 926, 945, 971, 1630
 Timeline Synchronisation Abbruch 911, 917, 924, 930, 934, 949, 957, 974, 1131, 1999
 Timeline Synchronisation der Timeline eines Elementes mit der Timeline des Elternelementes 885, 909, 915, 918, 922, 927, 932, 946, 954, 971, 1705
 Timeline Synchronisation Element mit Time-Container 881, 918, 931, 1623
 Timeline Synchronisation Time-Container mit Element 881, 918, 931, 1623
 Timeline synchronisieren 879, 911, 917, 924, 930, 934, 949, 957, 974, 1131, 1642, 1999
 Timeline Synchronisierung 868
 Timeline Synchronisierung der Animation des im Container liegenden Elementes auf Timeline 562, 573, 598, 615, 619, 628, 667, 682, 687, 692, 697, 700, 704, 709, 714, 718, 722, 726, 745, 766, 775, 909, 915, 922, 927, 946, 971, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1706
 Timeline Text laut BANNER ABSTRACT des aktiven Eintrages 913, 926, 940, 944, 970, 1576
 Timeline Text laut BANNER des aktiven Eintrages 913, 925, 940, 944, 970, 1575
 Timeline Text laut BANNER MOREINFOdes aktiven Eintrages 913, 926, 940, 944, 970, 1578
 Timeline Time Formate 872
 Timeline Time-Container Ende bezüglich seiner Kinder 881, 918, 931, 1623
 Timeline Time-Container erzeugen 868
 Timeline Timer dem Element zuordnen 876
 Timeline Timer für ein Objektgruppe erzeugen 868
 Timeline Timer für Objekt erzeugen 867
 Timeline Timer Implementierung 878
 Timeline Timer Neusetzung seiner Attribute 878
 Timeline Timer unverschachtelt 869
 Timeline Timer verschachtelt 870
 Timeline Timer verschachtelte synchronisieren 878
 Timeline Titel der Media-Datei 909, 915, 922, 927, 937, 946, 972, 1716
 Timeline Track 934, 937
 Timeline Track aktiv 934, 937
 Timeline Trackwechsel 917, 930, 949, 974, 1131, 1999
 Timeline Typ 709, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 955, 972
 Timeline Übergangsfiler 959
 Timeline Übergangsfiler Animationsschritte 966, 1730
 Timeline Übergangsfiler Ende-Prozentsatz 966, 1719
 Timeline Übergangsfiler Ergebnis der Animation 965, 1657
 Timeline Übergangsfiler Erscheinungsform des Überganges 966, 1703, 1723
 Timeline Übergangsfiler Gesamtdauer 965, 1619
 Timeline Übergangsfiler Interpolation 965, 1588
 Timeline Übergangsfiler Schrittweite 965, 1586
 Timeline Übergangsfiler Sichtbarkeit 965, 1657
 Timeline Übergangsfiler Start-Prozentsatz 965, 1629
 Timeline Übergangsfiler Unsichtbarkeit 965, 1657
 Timeline Übergangsfiler zeitliche Gesamtdauer 965, 1619
 Timeline unsichtbar machen eines Elementes 959
 Timeline unverschachtelt 869

Timeline unverschachtelter Timer 869
 Timeline Updates der Eigenschaften eines Elementes 909, 915, 922, 928, 946, 972, 1725
 Timeline Url der Media-Datei 909, 915, 922, 927, 937, 946, 971, 1695
 Timeline Url laut einem Kommando aus einer Advanced Streaming Format (ASF) Datei 917, 930, 1107, 1131, 1726, 2000
 Timeline Verfügbarkeit der Collection playList für Element auf der Timeline 914, 927, 945, 971, 1632
 Timeline Verlangsamung des Elementes auf der Timeline 885, 893, 897, 904, 908, 914, 918, 921, 926, 931, 945, 954, 965, 970, 1606
 Timeline verschachtelt 870
 Timeline verschachtelte synchronisieren 878
 Timeline verschachtelte Timelines synchronisieren 878
 Timeline verschachtelte Timer 870
 Timeline verschachtelte Timer synchronisieren 878
 Timeline Video wiedergeben 967
 Timeline visuellem Inhalt einer Media-Datei 908, 914, 921, 927, 945, 971, 1632
 Timeline vorhergehendes playItem Objekt aktivieren laut Collection playList 929, 933, 940, 947, 955, 1857
 Timeline Wartezeit für Start der Wiedergabe eines Media-Elementes 908, 914, 921, 927, 945, 971, 1649
 Timeline Wert eines Pixel-Intervalles zur Werterhöhung 893, 898, 904, 965, 1647
 Timeline Wert für Werterhöhung einer Style-Eigenschaft 893, 897, 904, 1584
 Timeline Wiedergabe per Media Bar-Player 1673
 Timeline Wiedergabe per Windows Media Player 1673
 Timeline Wiedergabe von Audio 911
 Timeline Wiedergabe von Video 967
 Timeline Wiedergabeplayer einem Media-Element zuordnen 908, 914, 921, 927, 945, 971, 1672
 Timeline Wiedergabeplayer Zeiger 909, 914, 922, 927, 945, 971, 1674
 Timeline Wiederholung der Animation des aktiven Elementes 887, 895, 900, 906, 911, 917, 919, 924, 930, 934, 948, 951, 957, 967, 974, 1126, 1994
 Timeline Zeiger auf das Eltern-Timeline 885, 894, 898, 905, 909, 915, 922, 927, 932, 946, 950, 955, 972, 1713
 Timeline Zeiger auf Wiedergabeplayer 909, 914, 922, 927, 945, 971, 1674
 Timeline zeitliche Toleranz für Zwangs-Synchronisation von Elementen auf der Timeline 909, 915, 918, 922, 927, 932, 946, 954, 971, 1707
 Timeline Zeitwert eines Pixel-Intervalles zur Werterhöhung 893, 898, 904, 965, 1648
 Timeline zurücksetzen 887, 895, 900, 906, 911, 917, 920, 924, 930, 934, 949, 951, 957, 974, 1127, 1994
 Timeline Zustand 866
 Timeline Zwangssynchronisierung 911, 917, 924, 930, 934, 948, 949, 957, 974, 1124, 1131, 1991, 1999
 Timeline Zwangs-Synchronisierung von Timelines 879
 Timelines synchronisieren 878
 Timeline-Steuerung 866
 Timeout 391, 1748
 Timer 390, 391, 1747, 1748
 Timer dem Element zuordnen 876
 Timer einer Rekursion 401, 1892
 Timer eines Scripts 401, 404, 1892, 1895
 Timer Konzept 865
 Timer Konzept Alternativen 866
 Timer Programmierung 865, 866
 Timer Rekursion 866



Timer rekursive Funktion 866
 Timer stoppen 390, 391, 1747, 1748
 Timeline Framenummer einer Media-Datei 908, 914, 921, 926, 945, 970, 1603
 Timeline Nummer des aktuellen Frames einer Media-Datei 908, 914, 921, 926, 945, 970, 1603
 Timeline pausieren 887, 895, 900, 906, 911, 917, 924, 930, 934, 948, 951, 957, 967, 974, 1125, 1992
 Titel 729
 Titel der Media-Datei 909, 915, 922, 927, 937, 946, 972, 1716
 Titel des Reset-Button 716
 Titelzeile 397, 440
 TITLE im HEAD 1359
 titlebar 397, 440
 TOBODY 1059, 1065, 1685
 TOC 874, 1469
 toGMTString() 489
 TOM 1001
 toolbar 397, 440
 Toolbar 397, 439, 440
 Toolbar persönliche 439
 Tooltip 571, 680, 686, 703, 981, 1172, 1564
 Tooltips 392, 408, 981, 1178, 1757, 1903
 Tooltip-Text 482, 563, 568, 573, 599, 610, 628, 649, 660, 667, 682, 688, 692, 697, 704, 709, 714, 718, 722, 727, 731, 740, 745, 754, 776, 1018, 1029, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1192, 1715
 Tooltip-Text Body 590, 615
 top 282, 397, 440, 986, 1029, 1033, 1037, 1043, 1048, 1053, 1059, 1066, 1071, 1709, 1727
 Top Margin Body 590, 1721
 topLeftClockwise 1704, 1723
 topLeftHorizontal 1704, 1723
 TOPMARGIN 584
 topToBottom 1703, 1723
 TR 811, 821, 822, 1056, 1350, 1935
 Track nächsten abspielen 928, 932, 940, 947, 955, 1847
 Track vorhergehenden abspielen 929, 933, 940, 947, 955, 1857
 Transformation linear 512, 2001
 Transition 749
 TRANSITIONTYPE 909, 915, 922, 928, 946, 972, 1721
 transparent anzeigen 811, 821, 1935
 transparenter Hintergrund 328, 990
 Transparenz 648, 666, 1564
 Transparenz IFRAME 665
 Trennstrich 987
 true 56, 57, 75, 76
 Trusted Sites 974
 try 193
 try catch 193
 try catch finally Anweisung 97, 128
 Typ der Objektvariable 67
 Typ der Timeline des Objektes 709, 885, 894, 898, 905, 909, 915, 922, 927, 937, 946, 950, 955, 972
 Typ der Verbindung 854, 1595
 Typ des Events 1772
 Typ Dokument 425, 1614
 Typ input Objekt 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 Typ Run-Time-Error 194, 1659
 type 638, 1169, 1171
 TYPE 309, 325, 451, 613, 634, 635, 636, 637, 638, 641, 670, 685, 756
 Type der Selektion 770, 1723

type="text/javascript" 46
 TypeError Run-Time-Error 194, 1659
 Typen 77, 142
 typeof 75
 typeof Operator 54, 77, 78, 81, 142
 typisierter Zeiger 61, 103
 Typographischer Effekt "Kashida" für Arabisch 815, 825, 1946
 tzeichenfolge 1184
 u0008 56, 58, 113, 114
 u0009 56, 58, 113, 114
 u000A 56, 58, 113, 114
 u000B 56, 58, 113, 114
 u000C 56, 58, 113, 114
 u000D 56, 58, 113, 114
 u0020 56, 57, 58, 113, 114
 u0022 56, 57, 58, 113, 114
 u0027 56, 57, 58, 113, 114
 u005C 56, 57, 58, 113, 114
 über mehrere Onlinesitzungen Daten speichern 862
 Übergabe von String-Werten zwischen Webseiten 562, 572, 1167, 1685
 übergeordnetes Fenster laut Fensterhierarchie 221, 383, 1667
 Überlagerung Fenster 440
 überlappende Objekte 772
 überlappende Textbereiche 980
 Überlappung von IFRAME 666
 Übernahme der Ereigniskontrolle 361, 1145
 Überschreiben des onXXX-Ereignis-Handlers 291
 Überschrift 984, 986
 Überschrift im Dialogfenster 981
 Überschrift in der aktiven Fenstertitelzeile 981
 Überschrift mit permanentem Farbwechsel 1928
 Überschrift nichtaktive Fenstertitelzeile 981
 Übersichtlichkeit im Quellcode 48, 61
 Übersichtlichkeit Scriptcode 48, 61
 Uhrzeitformat 158, 212, 1910
 Umbruch des Inhaltes des HTML-Elementes 992
 Umfluss Body 589, 614
 Umfluss um ein Objekt 481, 589, 609, 614, 627, 1581
 Umflussrichtung 425, 481, 561, 571, 597, 609, 618, 627, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 739, 743, 753, 774, 1016, 1028, 1032, 1036, 1041, 1046, 1051, 1077, 1191, 1610
 Umflussrichtung Body 589, 614, 759, 765
 Umgebung eines Objektes und Abstand Aussenrand 589, 605, 774, 796, 803, 811, 812, 813, 821, 822, 823, 976, 1582, 1650, 1654, 1682, 1721, 1935, 1936, 1938, 1939
 Umgebungseinstellung als Cookie 485
 Umgebungsobjekt 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 714, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1685
 Umschalt 1105
 Umwandlung eines Url-Wertes 337, 780, 828
 unadorned 406, 409
 UnBookmark 278, 477
 Und 75
 undefined 54, 77, 78, 81, 142
 underline 987, 990
 Underline 278, 477
 Unendlich 204, 1660
 unescape() 143, 317, 324, 647, 657



Unicode Dateneingabe bidirektional 816, 818, 825, 1947
 Unicode 57, 113, 217, 988, 1775
 Unicode-Format 144, 153, 211, 213, 1766
 Uniform Resource Identifier 648, 666, 681, 1652
 Uniform Resource Identifiers 144, 153, 211, 213, 1766
 Uniform Resource Name 239, 247, 482, 504, 507, 563, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1709, 1726, 1734
 Uniform Ressource Identifier 152, 153, 1758, 1764
 uninitialized 246, 428, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1195, 1620, 1679
 Universal Coordinated Time 176
 Universal-Bereich erzeugen Selektion 458, 770, 1083, 1757
 UniversalBrowserWrite 283, 361, 1145, 1146
 Unix 412
 Unlink 278, 477
 Unload eines Dokumentes 1114, 1131, 1979, 1999
 Unselect 278, 477
 UNSELECTABLE 482, 563, 568, 573, 590, 599, 610, 615, 619, 628, 649, 660, 667, 682, 688, 692, 697, 701, 705, 709, 714, 718, 723, 727, 731, 745, 754, 761, 776, 1018, 1029, 1042, 1048, 1053, 1065, 1078, 1192, 1725
 Unterschrift zu einer Microsoft Active Channel-Datei 1164, 1830
 Unterverzeichnisse 50
 up 483, 1080, 1610, 1763
 Update der Browserversion 410, 1169, 1566
 uppercase 987, 990
 URI 144, 152, 153, 211, 213, 648, 666, 681, 1652, 1758, 1764, 1766
 URIError Run-Time-Error 194, 1659
 Url 561, 572, 736, 1167, 1633
 URL 1108
 Url als Anker 561, 572, 1166, 1631
 Url Autocomplete 1163, 1742
 Url Autovervollständigung 1163, 1742
 Url Dateiname ohne Path und ohne Protokoll liefern 681, 1660
 Url der Daten 568, 649, 667, 682, 704, 1188, 1195, 1694
 Url der Daten des Objektes 601, 1603
 Url der direkt vorhergehenden Seite 428, 1679
 Url der Media-Datei 909, 915, 922, 927, 937, 946, 971, 1695
 Url des Dokumentes 396, 430, 626, 1558, 1726
 Url des Image in geringerer Auflösung 681, 703, 1654
 Url des Servers und des dortigen Dokumentes 626, 1558
 Url einer CSS-Datei 996, 1634
 Url eines Behaviors im Objekt Style 337, 781
 Url mit angehängten Textdaten 317, 323, 646, 657
 Url Protokoll-Teil inklusive http und ftp liefern 428, 562, 572, 682, 1677
 Url von Videoclip 681, 686, 703, 1620
 url() 337, 781, 828, 1912, 1951
 URL-Eingabezeile 397, 439
 Url-Wert erzeugen 337, 780, 828
 Url-Wert umwandeln 337, 780, 828
 URN 239, 247, 482, 504, 507, 563, 568, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 731, 736, 740, 745, 750, 751, 754, 761, 767, 776, 1018, 1029, 1033,

1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1709, 1726, 1734
 USEMAP 325, 670, 738
 User Informationen 1167
 User Online-Zustand 410
 User Sprache des Betriebssystems 1170, 1727
 User Surfgewohnheiten 485
 User Visitenkarte 1731
 userAgent 1168
 User-Agent 412, 1726
 user-agent: * 49
 user-agent: WebCrawler 49
 User-Cache 976, 1624, 1837, 1874
 UserDataStore 976, 1624, 1837, 1874
 Userdaten 1731
 Userdaten cachen 974
 Userdaten permanent sichern 976, 1624, 1837, 1874
 Userdaten permanent speichern 976, 1624, 1837, 1874
 User-Daten permanent speichern 1162, 1163, 1742
 User-Daten speichern 1162, 1163, 1742
 User-Daten versteckt 622
 Userdaten verwalten 976, 1624, 1837, 1874
 Usereingabe im Formular 725
 Userprofil 1731
 User-Profil lesen 1172, 1778
 User-Profil vCard-Wert lesen 1172, 1778
 userProfile 1360
 User-Profile-Informationen 1171
 Ursprung des Grafiksystems 1182
 UTC 176, 873
 UTC-Zeitformat 975
 UTF-8 Zeichen 152, 153, 1764
 VALIGN 811, 816, 820, 825, 1933, 1947
 value 638
 VALUE 248, 466, 599, 633, 635, 637, 638, 641, 686, 688, 693, 697, 701, 705, 709, 714, 716, 718, 723, 727, 761, 767, 1078, 1572, 1727
 valueOf Operator 143
 var Anweisung 63, 67, 69, 99
 var ie = document.all ? true : false; 136
 var JScript-Objekt 149
 var ns = document.layers ? true : false; 136
 var Objekt 99, 149, 160, 1190
 var Objekt Events 1142, 1963
 var Objekt Filter 555
 var Objekt Styles 847
 var Script-Objekt 63, 160
 Variable 62
 Variable aus String-Literal 213
 Variable Bezeichner 62
 Variable Datentyp 63
 Variable Deklaration 63
 Variable globale Gültigkeit 65
 Variable Gültigkeit 65
 Variable Gültigkeit global 65
 Variable Gültigkeit in Funktion 95, 105
 Variable Gültigkeit lokal 65
 Variable in Funktion 95, 105
 Variable Instanziierung 64
 Variable lokale Gültigkeit 65
 Variable Löschung 65
 Variable mit konstantem Inhalt 101
 Variable Name 62
 Variable nicht per new erzeugt 66
 Variable Referenzierung 64
 Variable Speicherplatz 65, 143



Variable String 213
 Variable und ihre Komponenten 62
 Variable Wertzuweisung 62
 Variable Zugriff 62
 Variable: nicht Objektvariable 66
 Variablen 52
 Variablenexistenz 64
 Variablenoperatoren 75
 Variablenübergabe 113
 Variablenzugriff 62, 63, 64, 65
 Varinate input Objekt 688, 692, 697, 700, 704, 709, 714, 718, 722, 727, 1721
 VBA 40
 VBArray Objekt 1428, 1429, 1833, 1835
 vbs 1649
 vbscript 1649
 VBScript 40, 402, 404
 VB-Script 1723
 vCard.Business.City 1573
 vCard.Business.Country 1573
 vCard.Business.Fax 1573
 vCard.Business.Phone 1573
 vCard.Business.State 1573
 vCard.Business.StreetAddress 1573
 vCard.Business.URL 1573
 vCard.Business.Zipcode 1573
 vCard.Cellular 1573
 vCard.Company 1573
 vCard.Department 1573
 vCard.DisplayName 1573
 vCard.Email 1573
 vCard.FirstName 1573
 vCard.Gender 1573
 vCard.Home.City 1573
 vCard.Home.Country 1573
 vCard.Home.Fax 1573
 vCard.Home.Phone 1573
 vCard.Home.State 1573
 vCard.Home.StreetAddress 1573
 vCard.Home.Zipcode 1573
 vCard.Homepage 1573
 vCard.JobTitle 1573
 vCard.LastName 1573
 vCard.MiddleName 1573
 vCard.Notes 1573
 vCard.Office 1573
 vCard.Pager 1573
 vCard-Daten 1171
 vCard-Schema 1573
 vCard-Wert lesen 1172, 1778
 Veränderbarkeit der Fenstergröße 303
 verändern 1096
 verändern Fenstergröße 399, 1872
 verändern HTML-Element 1096
 Veränderung des HTML-Elementes 1103
 Verbindung Offlinezustand 854, 1595
 Verbindung Onlinezustand 854, 1595
 Verbindung sichere 489
 Verbindung Typ 854, 1595
 Verfallsdatum Cookie 489
 Verfügbarkeit Data Tainting 412, 1909
 verfügbare Breite des Arbeitsbereiches 854, 1574
 verfügbare Farben 1182
 verfügbare Höhe des Arbeitsbereiches 854, 1574
 Verfügbarkeit der Microsoft Virtual Machine 854, 1647
 Verfügbarkeit Java 412, 1834

Vergleich auf Identität in Wert und Datentyp 78
 Vergleich auf Wert und Datentyp 78
 Vergleich Datentyp und Wert 78
 Vergleich von Objektinstanzen 143
 Vergleich von Zeigern 143
 Vergleich zweier Objektinstanzen 149
 Vergleich zweier String-Literale 220, 1838
 Vergleich zweier Strings 220, 1838
 vergleichende Operatoren 76
 Vergleichoperatoren 76
 Vergleichsoperatoren 78
 Verhalten eines Elementes 809, 819, 1919
 Verhalten Element 507, 1761
 Verhalten Objekt 507, 1761
 Verhalten Standard 809, 819, 1919
 Verhalten Tag 507, 1761
 Verhaltensweisen von Elementen 490, 504
 verkettende Operatoren 76
 Verkettung von String-Literalen 216, 1753
 Verkettung von String-Objekten 216, 1753
 Verkettung von Zeichenketten 76
 Verkettungsoperator 214
 Verkettungsoperatoren für Zeichenketten 76
 Verknüpfungen 1445
 Verkürzung der Punktnotation 48, 61
 verlassen 1095
 Verlassen 1097
 Verlassen der HTML-Seite 1097
 verlassen Dokument 1104, 1129, 1996
 verlassen eines Dokumentes 1114, 1979
 verlassen HTML-Dokument 1104
 verlassen HTML-Element 1095
 verlassen HTML-Seite 1104
 verlassen Seite 1104
 Verlauf 281, 283, 370, 382, 437, 863, 1167, 1633, 1739, 1869
 Verlauf des Surfens 1165
 Vermeidung von Cookies 486
 verschachtelten Elemente 1103
 Verschachtelung Funktion 88, 106
 verschieben Dokument im Fenster 400, 401, 1876
 verschieben Fenster 1103
 Verschiebung von Bits 79, 80
 Verschlüsselung anhand von Zertifikaten 284
 Version 1.0 30
 Version 1.1 30
 Version Browser 1170, 1567, 1726
 Version der Jscript-Scriptmaschine 100
 Version des Browsers 17
 Version Scriptmaschine 17
 Version von Javascript 17
 Versionsnummer 1167
 Versionsnummer Browser 410, 1566
 Versionsprüfung Komponente 567, 753, 1594
 versteckt User-Daten 622
 vertical 1703, 1723
 vertikale Bildpunkte 1183, 1608, 1652
 vertikale Linie 325, 670
 vertikaler Scrollbereich 482, 562, 568, 590, 598, 610, 615, 628, 663, 682, 687, 692, 696, 704, 709, 713, 718, 722, 726, 731, 744, 750, 754, 760, 766, 775, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1684
 vertikaler Tabulator 56, 58, 113, 114, 1184
 verwalten einer Collection 188
 Verwaltung Cookies 1167



Verwendung von Cookies 484
 Verzeichnis lesen 1559, 1656
 Video 563, 737, 754, 865, 866, 874, 887, 907, 911, 917, 919, 924, 930, 934, 948, 957, 967, 973, 1120, 1121, 1256, 1468, 1721, 1987, 1988
 Video Startzeitpunkt festlegen 1256
 Video Tooltip 571, 680, 686, 703, 1564
 Video wiedergeben 967
 video/mpeg 1556
 video/x-ms-asf 874, 1468, 1470
 video/x-ms-asf' 858, 1568
 video/x-ms-wmv 874, 1468
 video/x-ms-wvx 1470
 Videoclip 671, 681, 686, 703, 1620
 Videoclip starten 682, 686, 704, 1696
 Videoclip Url 681, 686, 703, 1620
 Videoclip-Sound 575
 Videos 865, 887
 Vieleck 1688
 Viereck 1688
 Viewbereich Objekt 457, 460, 484, 564, 570, 575, 601, 603, 612, 617, 621, 631, 669, 684, 690, 694, 699, 706, 711, 716, 720, 724, 729, 733, 742, 747, 756, 763, 771, 778, 1027, 1031, 1035, 1039, 1045, 1050, 1055, 1062, 1068, 1074, 1081, 1085, 1194, 1876
 Virtual Reality Modeling Language 671
 Virtuelle Javamaschine 566
 Virtuelle Maschine 566
 virtueller Host 24
 visibility 528, 672, 1712
 VISIBILITY 327
 Visitenkarte 1731
 Visual Basic 224
 VisualBasic-Array nach JScript-Array konvertieren 171
 VLINK Farbe 590, 1731
 VM 566
 void 1016, 1627
 void Operator 77, 78, 81
 Vollbild 397, 439
 VOLUME 576
 vom Browser erzeugter Zeiger 59
 vom Browser geparkt 222
 vom Browser interpretiert 222
 vom Druck ausschließen 420, 982, 1679
 vordefiniert 58
 vordefinierte Bezeichner 980
 vordefinierte Dimensionen 980
 vordefinierte Farbbereiche 981
 vordefinierte Funktion 106
 vordefinierte Kommandos 275, 474
 vordefinierte Objekte zum Browser 220
 vordefinierte Objekte: Hierarchie 220
 vordefinierte Operationen mit Objekten 142
 vordefinierte Schriften 980
 vordefinierte Werte für Attribute 52
 vordefiniertes Objekt 68, 106
 Vordergrundfarbe 277, 426, 476, 810, 820, 1625, 1928
 Vordergrundfarbe Body 590, 1711
 Vordergrundfarbe im Dokument 426, 1625
 Vorgängerkind 246, 562, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 760, 766, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1077, 1192, 1675
 vorhergehende Seite laut previous laden 1165
 vorhergehende Seite Url 428, 1679

vorhergehenden Track abspielen 929, 933, 940, 947, 955, 1857
 VORHERGEHENDES Kind 241
 vorladen Bild 675
 vorladen Grafik 675
 Vornull bei Ziffern-Zeichenketten 1908
 Vorwärts-Button 863, 1775
 VRML 369, 681, 703, 1169, 1170, 1620
 VRML-Datei 671
 VRML-Datei starten 682, 686, 704, 1696
 vsides 1016, 1627
 vspace 671
 VSPACE 306, 325, 450, 566, 570, 670, 671
 vzeichenfolge 1184
 W3C 801
 Wagenrücklauf 56, 58, 113, 114, 1184
 wait 988, 993
 Wave Filter 546
 Web auf dem Server 49
 WebCrawler darf ALLES außer /test/ scannen 50
 WebCrawler darf ALLES außer /test/test.htm scannen 50
 WebCrawler darf ALLES scannen 50
 WebCrawler darf NICHTS scannen 50
 Web-Projekt 51
 Web-Projekte 223
 Webseite als Datei speichern 1129, 1996
 Webseite Autoscan einer Url 1163, 1742
 Webseite Festplattenordner 24
 Webseite Host 24
 Webseite mit eigenem Icon ab IE 5.x 419, 1634, 1679
 Webseite nicht gefunden 1163, 1742
 Webseite per Apache-HTTP-Server 25
 Webseite Root 24
 Webseite Suche 1162
 Webseite Textsuche 1164, 1844
 Webseite verlassen 1129, 1996
 Webseite zum Microsoft Active Desktop hinzufügen 1162, 1737
 Webseiten Suche 562, 572, 1167, 1685
 Webseiten Übergabe von String-Werten 562, 572, 1167, 1685
 Webspeech aktive Sprachausgabe 1515
 Webspeech Anzeige des Textes 1517
 Webspeech AUTHKEY 1507
 Webspeech AUTOSTART 1510
 Webspeech BACKGROUNDCOLOR 1511
 Webspeech Bedienung durch User 1511
 Webspeech beenden der Sprachausgabe 1519
 Webspeech Beispiel zur Javascript-Programmierung 1519
 Webspeech Control-Elemente-Leiste 1511
 Webspeech CONTROLPOSITION 1511
 Webspeech Deckkraft der Hintergrundfarbe 1517
 Webspeech DEFAULT 1510
 Webspeech Eigenschaften der Sprachausgabe 1513
 Webspeech Fontheöhe der Textanzeige 1518
 Webspeech Haupt-Version 1513
 Webspeech Hintergrundfarbe Deckkraft 1517
 Webspeech Hintergrundfarbe 1511, 1516
 Webspeech IMMEDIATE 1510
 Webspeech Lauftext 1510
 Webspeech Lippen 1510
 Webspeech Lizenzschlüssel 1498, 1507
 Webspeech Lizenz-Schlüssel 1516
 Webspeech LOGOX-Datei 1509
 Webspeech MOUTHANIMATION 1510



Webspeech MOUTHCOLOR 1510
 Webspeech Mundanimation 1516
 Webspeech Mundbereich 1510
 Webspeech Mundfarben 1516
 Webspeech OPAQUE 1511
 Webspeech pausieren der Sprachausgabe 1515
 Webspeech pausierte Sprachausgabe 1514
 Webspeech pausierte Sprachausgabe fortsetzen 1515
 Webspeech Position Textanzeige 1518
 Webspeech Pufferung von Text 1514, 1515, 1517
 Webspeech Rachen 1510
 Webspeech Sprachausgabe aktiv 1515
 Webspeech Sprachausgabe beenden 1519
 Webspeech Sprachausgabe Eigenschaften 1513
 Webspeech Sprachausgabe fortsetzen bei Pause 1515
 Webspeech Sprachausgabe pausieren 1514, 1515
 Webspeech Sprachausgabe starten 1518, 1519
 Webspeech Sprachausgabe stummschalten 1514
 Webspeech starten der Sprachausgabe 1518, 1519
 Webspeech Steuerungselemente 1516
 Webspeech stummgeschaltete Sprachausgabe 1514
 Webspeech TEXT 1509, 1510
 Webspeech Text anzeigen 1517
 Webspeech Text puffern 1517
 Webspeech TEXTANIMATION 1510
 Webspeech Textanzeige Fonthöhe 1518
 Webspeech Textanzeige Position 1518
 Webspeech TEXTCOLOR 1510
 Webspeech Textfarben 1518
 Webspeech Textfonthöhe 1511
 Webspeech TEXTPOSITION 1511
 Webspeech Text-Pufferung 1514, 1515
 Webspeech TEXTSIZE 1511
 Webspeech TEXTURE 1510
 Webspeech TRANSP 1510
 Webspeech TRANSPARENT 1510
 Webspeech Transparenz 1511
 Webspeech URL 1509
 Webspeech Version 1514
 Webspeech von Logox 1498
 Webspeech Zähne 1510
 Webspeech Zunge 1510
 Webspeech-Objekt erzeugen 1500
 Webspeech-Objekt in Javascript erkennen 1511
 Webspeech-Objekt in Javascript erzeugen 1511
 Webspeech-Objekt in Javascript programmieren 1513
 Webspeech-Objekt in Javascript verwalten 1511
 Webspeech-Objekt und Events 1519
 Webspeech-Objekt und seine Methoden 1513
 Webspeech-Sprechtags 1528
 Wechsel zu anderer HTML-Seite 1104
 Wechsel zu anderer Seite 1104
 Wechselnder Text in Statuszeile 287, 385, 1700
 weiterreichen des Events 300
 Weltzeit 176, 873
 Wert auf nicht-numerisch prüfen 206
 Wert auf numerisch endlich prüfen 205
 Wert des Argumentes 161, 1553
 Wert des Kindes 238
 Wert des Knoten 238
 Wert des STYLE-Attributes im Tag des HTML-
 Elementes 810, 817, 820, 1929
 Wert einem Eingabefeld zuweisen 640
 Wert einer Style-Eigenschaft 240, 241, 253, 257, 261,
 264, 483, 484, 564, 565, 570, 574, 575, 592, 593, 600,
 601, 611, 612, 613, 617, 618, 620, 621, 622, 629, 630,

631, 668, 669, 683, 684, 689, 690, 693, 694, 695, 698,
 699, 701, 702, 706, 707, 710, 711, 712, 715, 716, 719,
 720, 721, 724, 725, 728, 729, 733, 734, 746, 747, 755,
 756, 762, 763, 768, 769, 777, 778, 826, 827, 828, 849,
 850, 1021, 1027, 1030, 1031, 1032, 1034, 1035, 1038,
 1039, 1044, 1045, 1049, 1050, 1051, 1054, 1055, 1056,
 1061, 1062, 1067, 1068, 1073, 1074, 1081, 1082, 1193,
 1194, 1195, 1770, 1787, 1867, 1888
 Wert eines Attributes 238
 Wert eines Attributes belegen 262, 484, 565, 570, 575,
 593, 601, 603, 612, 618, 621, 651, 662, 665, 669, 684,
 690, 695, 699, 702, 707, 711, 716, 720, 725, 729, 734,
 738, 742, 747, 750, 756, 763, 769, 778, 828, 849, 850,
 863, 864, 1027, 1031, 1035, 1039, 1045, 1050, 1056,
 1062, 1068, 1074, 1082, 1172, 1190, 1194, 1886
 Wert eines Cookies 488
 Wert eines HTML-Attributes aus Funktionsaufruf 103
 Wert eines JScript-Objektes ermitteln 159, 213, 1913
 Wert eines Objekt-Attributes 247, 599, 686, 688, 697,
 701, 705, 709, 714, 718, 723, 727, 761, 767, 1078,
 1727
 Wert eines Objektes in einen String umwandeln 148,
 212, 1911
 Wert eines Objektes in System-lokale Einstellungen
 umwandeln 158, 212, 1910
 Wert eines per HTML-Attributes 254, 483, 563, 569,
 574, 591, 599, 602, 611, 616, 620, 629, 650, 661, 664,
 668, 683, 689, 693, 698, 701, 705, 710, 715, 719, 723,
 728, 732, 737, 740, 746, 750, 755, 761, 768, 777, 827,
 849, 850, 863, 864, 1021, 1030, 1034, 1038, 1044,
 1049, 1054, 1060, 1066, 1072, 1080, 1189, 1193, 1777
 Wert eines Zeigers 70
 Wert in Exponential-Darstellung 205, 1909
 Wert in Festkomma-Darstellung 205, 1909
 Wert konvertieren 206
 Wert konvertieren nach Floating-point 206
 Wert konvertieren nach Integer 207
 Wert konvertieren nach String 208
 Wert liefern 208
 Wert maximal in Script 204
 Wert minimal in Script 204
 Wert nach Boolean konvertieren 152, 1745
 Wert Nachkommastellen 205
 Wert negativ unendlich 204
 Wert nicht numerisch 204
 Wert positiv unendlich 205
 Wert und Datentyp 78
 Wert vom Attribut 262, 484, 564, 570, 575, 593, 601,
 603, 612, 618, 621, 651, 662, 665, 669, 684, 690, 695,
 699, 702, 707, 711, 716, 720, 725, 729, 734, 738, 742,
 747, 750, 756, 763, 769, 778, 828, 849, 850, 863, 864,
 1027, 1031, 1035, 1039, 1045, 1050, 1056, 1062, 1068,
 1074, 1082, 1172, 1190, 1194, 1886
 Wert von Attribut 238
 Werte der Argumente aus dem Funktionsaufruf einlesen
 104, 161, 162, 1553, 1650
 Werte von Attributen 222
 Wertekonstanten 56
 Werte-Vergleich 76
 Wertkonvertierung einer Url beim Objekt Style 337,
 781, 828, 1912, 1951
 Wertzuweisung auf Variable 62
 Wheel Filter 546
 while Anweisung 86, 99
 wichtige Objekte - Übersicht 1221
 Widergabe per Media Bar-Player 1673



Wiedergabe per Windows Media Player 1673
 Wiedergabegeschwindigkeit 885, 890, 893, 898, 905, 909, 915, 922, 927, 932, 946, 954, 966, 971, 1693
 Wiedergabe-Lautstärke 583, 1731
 Wiedergabeplayer für Medien 908, 909, 914, 921, 922, 927, 945, 971, 1672, 1674
 widow 987
 width 397, 440, 671
 WIDTH 306, 309, 325, 450, 451, 566, 570, 613, 670, 671, 756
 Wiederholung 893, 898, 904, 909, 915, 922, 927, 932, 945, 954, 965, 971, 1679, 1686, 1690
 Wiederholungen Hintergrundmusik 582, 681, 703, 1653
 Wiederholungen Sound 582, 681, 703, 1653
 Win16 412, 1167, 1672
 Win32 412, 1167, 1672
 WinCE 412, 1672
 window 282, 1103, 1360
 window.navigator.mimeTypeCollection Collection des Netscape 368
 window Objekt 281, 383, 388, 419, 1592, 1593, 1594, 1607, 1608, 1609, 1610, 1614, 1623, 1627, 1633, 1651, 1659, 1660, 1662, 1666, 1667, 1682, 1684, 1687, 1699, 1721, 1738, 1741, 1745, 1747, 1748, 1750, 1753, 1757, 1760, 1769, 1775, 1839, 1842, 1844, 1849, 1859, 1872, 1876, 1886, 1892, 1895, 1901, 1903
 window Objekt des Internet Explorer 370
 window Objekt des Netscape 283
 window Objekt Events 1142, 1963
 window Objekt Filter 556
 window Objekt Styles 847
 window.captureEvents() 360, 361, 1144, 1145
 window.clientInformation Objekt 378, 412, 1566, 1567, 1583, 1592, 1598, 1601, 1665, 1672, 1708, 1726, 1727, 1834, 1909
 window.clientInformation Objekt des Internet Explorer 410
 window.clipboardData Objekt des Internet Explorer 413
 window.close() 440
 window.dialogArguments Objekt 378, 1608
 window.dialogArguments Objekt des Internet Explorer 416
 window.document Objekt des Internet Explorer 418
 window.document.anchors Collection des Netscape 306
 window.document.applets Collection des Netscape 306
 window.document.body Objekt des Netscape 312
 window.document.cookie Collection des Netscape 307
 window.document.embeds Collection des Netscape 309
 window.document.forms Collection des Netscape 309
 window.document.frame Objekt des Netscape 312
 window.document.frames Collection des Netscape 310
 window.document.frameSet Objekt des Netscape 318
 window.document.iLayer Objekt des Netscape 326
 window.document.images Collection des Netscape 310
 window.document.img Objekt des Netscape 324
 window.document.iLayer Objekt des Netscape 326
 window.document.layers Collection des Netscape 310
 window.document.link Objekt des Netscape 332
 window.document.links Collection des Netscape 311
 window.document.plugins Collection des Netscape 311
 window.document.span Objekt des Netscape 333
 window.document.style Objekt des Netscape 334
 window.document.styleSheet Objekt des Netscape 334
 window.enableExternalCapture(); 361, 1145
 window.event Objekt des Netscape 353

window.external Objekt 381, 1162, 1625, 1656, 1737, 1738, 1742, 1827, 1831, 1844, 1901
 window.history Objekt des Netscape 366
 window.location Objekt des Netscape 366
 window.location.hash 306, 449, 565
 window.location.hostname 25
 window.location.href 306, 395, 449, 565, 1844
 window.navigator Objekt des Netscape 367
 window.navigator.mimeTypeCollection Collection des Netscape 369
 window.navigator.plugins Collection des Netscape 368, 369
 window.onload 503, 850, 858, 865, 1981
 window.popup Objekt 392, 1172, 1181, 1182, 1614, 1643, 1757, 1899
 window.screen Objekt des Netscape 370
 window.status 287, 384, 1143, 1700
 Window-Clipboard 1103
 window-Events 1208, 1964
 Window-Name logisch 437
 Window-Name physisch 437
 Window-Objekt 1096, 1097
 Window-Objekt Ereignisse 298
 Windows 16-Bit 412, 1672
 Windows 32-Bit 412, 1672
 Windows 98 und JScript 18, 31, 42, 135
 Windows 98 und Scriptmaschine 18, 31, 42, 135
 Windows ab NT 5.x 224
 Windows Bitmap 671, 702
 Windows CE 412, 1672
 Windows Enhanced Metafile 671, 702
 Windows Explorer starten 1452
 Windows Media Datei 875, 1469
 Windows Media Player 858, 859, 860, 1119, 1124, 1126, 1130, 1621, 1666, 1675, 1986, 1991, 1993, 1998
 Windows Media Player 7.1 575, 873, 1460
 Windows Media Player 7.1 .add() 1493, 2013
 Windows Media Player 7.1 .appendItem() 1490, 2013
 Windows Media Player 7.1 .attributeCount 1488, 1490, 2009
 Windows Media Player 7.1 .attributeName() 1490, 2013
 Windows Media Player 7.1 .autoStart 1497, 2009
 Windows Media Player 7.1 .balance 1497, 2009
 Windows Media Player 7.1 .bandWidth 1493, 2009
 Windows Media Player 7.1 .baseURL 1497, 2009
 Windows Media Player 7.1 .bitRate 1493, 2009
 Windows Media Player 7.1 .bufferingCount 1493, 2009
 Windows Media Player 7.1 .bufferingProgress 1493, 2009
 Windows Media Player 7.1 .bufferingTime 1493, 2009
 Windows Media Player 7.1 .captioningID 1484, 2009
 Windows Media Player 7.1 .cdromCollection 1478, 2009
 Windows Media Player 7.1 .clearErrorQueue() 1491, 2013
 Windows Media Player 7.1 .close() 1480, 2013
 Windows Media Player 7.1 .closedCaption 1478, 2009
 Windows Media Player 7.1 .controls 1478, 2009
 Windows Media Player 7.1 .count 1484, 1490, 1493, 1496, 2009
 Windows Media Player 7.1 .currentItem 1485, 2009
 Windows Media Player 7.1 .currentMarker 1478, 1485, 2009
 Windows Media Player 7.1 .currentMedia 2009
 Windows Media Player 7.1 .currentPlaylist 1478, 2010
 Windows Media Player 7.1 .currentPosition 1485, 2010



Windows Media Player 7.1 .currentPositionString 1485, 2010
 Windows Media Player 7.1 .defaultFrame 1497, 2010
 Windows Media Player 7.1 .downloadProgress 1493, 2010
 Windows Media Player 7.1 .driveSpecifier 1484, 2010
 Windows Media Player 7.1 .duration 1488, 2010
 Windows Media Player 7.1 .durationString 1488, 2010
 Windows Media Player 7.1 .eject() 1484, 2013
 Windows Media Player 7.1 .enableContextMenu 1478, 2010
 Windows Media Player 7.1 .enabled 1478, 2010
 Windows Media Player 7.1 .enableErrorDialogs 1497, 2010
 Windows Media Player 7.1 .encodedFrameRate 1494, 2010
 Windows Media Player 7.1 .error 1478, 2010
 Windows Media Player 7.1 .errorCount 1491, 2010
 Windows Media Player 7.1 .errorDescription 1491, 2010
 Windows Media Player 7.1 .fastForward() 1485, 2013
 Windows Media Player 7.1 .fastReverse() 1485, 2013
 Windows Media Player 7.1 .FrameRate 1494, 2010
 Windows Media Player 7.1 .framesSkipped 1494, 2010
 Windows Media Player 7.1 .fullScreen 1478, 2010
 Windows Media Player 7.1 .getAll() 1493, 1496, 2013
 Windows Media Player 7.1 .getAttributeName() 1488, 2013
 Windows Media Player 7.1
 .getAttributeStringCollection() 1493, 2013
 Windows Media Player 7.1 .getByAlbum() 1493, 2013
 Windows Media Player 7.1 .getByAttribute() 1493, 2013
 Windows Media Player 7.1 .getByAuthor() 1493, 2013
 Windows Media Player 7.1 .getByDriveSpecifier() 1484, 2013
 Windows Media Player 7.1 .getByGenre() 1493, 2013
 Windows Media Player 7.1 .getByName() 1493, 1496, 2013
 Windows Media Player 7.1 .getItemInfo() 1488, 1490, 2014
 Windows Media Player 7.1 .getItemInfoByAtom() 1488, 2014
 Windows Media Player 7.1 .getMarkerName() 1488, 2014
 Windows Media Player 7.1 .getMarkerTime() 1488, 2014
 Windows Media Player 7.1 .getMediaAtom() 1493, 2014
 Windows Media Player 7.1 .getMode() 1498, 2014
 Windows Media Player 7.1 .getProxyBypassForLocal() 1494, 2014
 Windows Media Player 7.1 .getProxyExceptionList() 1494, 2014
 Windows Media Player 7.1 .getProxyName() 1494, 2014
 Windows Media Player 7.1 .getProxyPort() 1494, 2014
 Windows Media Player 7.1 .getProxySettings() 1494, 2014
 Windows Media Player 7.1 .imageSourceHeight 1488, 2010
 Windows Media Player 7.1 .imageSourceWidth 1488, 2010
 Windows Media Player 7.1 .importPlaylist() 1496, 2014
 Windows Media Player 7.1 .insertItem() 1490, 2014
 Windows Media Player 7.1 .invokeURLs 1497, 2010
 Windows Media Player 7.1 .isAvailable() 1486, 1498, 2014

Windows Media Player 7.1 .isDeleted() 1493, 1496, 2014
 Windows Media Player 7.1 .isIdentical() 1488, 1490, 2014
 Windows Media Player 7.1 .isMemberOf() 1488, 2014
 Windows Media Player 7.1 .isOnline 1478, 2011
 Windows Media Player 7.1 .isReadOnlyItem() 1488, 2014
 Windows Media Player 7.1 .item() 1484, 1490, 1491, 1493, 1496, 2014
 Windows Media Player 7.1 .launchURL() 1480, 2014
 Windows Media Player 7.1 .lostPackets 1494, 2011
 Windows Media Player 7.1 .markerCount 1488, 2011
 Windows Media Player 7.1 .maxBandwidth 1494, 2011
 Windows Media Player 7.1 .mediaCollection 1479, 2011
 Windows Media Player 7.1 .moveItem() 1490, 2014
 Windows Media Player 7.1 .mute 1498, 2011
 Windows Media Player 7.1 .name 1488, 1490, 2011
 Windows Media Player 7.1 .network 1479, 2011
 Windows Media Player 7.1 .newPlaylist() 1496, 2014
 Windows Media Player 7.1 .next() 1486, 2014
 Windows Media Player 7.1 .openState 1479, 2011
 Windows Media Player 7.1 .pause() 1486, 2014
 Windows Media Player 7.1 .play() 1486, 2015
 Windows Media Player 7.1 .playCount 1498, 2011
 Windows Media Player 7.1 .playItem() 1486, 2015
 Windows Media Player 7.1 .playlist 1484, 2011
 Windows Media Player 7.1 .playlistCollection 1479, 2011
 Windows Media Player 7.1 .playState 1479, 2011
 Windows Media Player 7.1 .previous() 1486, 2015
 Windows Media Player 7.1 .rate 1498, 2011
 Windows Media Player 7.1 .receivedPackets 1494, 2011
 Windows Media Player 7.1 .receptionQuality 1494, 2012
 Windows Media Player 7.1 .recoveredPackets 1494, 2012
 Windows Media Player 7.1 .remove() 1493, 1496, 2015
 Windows Media Player 7.1 .removeItem() 1490, 2015
 Windows Media Player 7.1 .SAMIFilename 1484, 2012
 Windows Media Player 7.1 .SAMILang 1484, 2012
 Windows Media Player 7.1 .SAMISStyle 1484, 2012
 Windows Media Player 7.1 .setDeleted() 1493, 1496, 2015
 Windows Media Player 7.1 .setItemInfo() 1488, 1491, 2015
 Windows Media Player 7.1 .setMode() 1498, 2015
 Windows Media Player 7.1 .setProxyBypassForLocal() 1494, 2015
 Windows Media Player 7.1 .setProxyExceptionList() 1494, 2015
 Windows Media Player 7.1 .setProxyName() 1494, 2015
 Windows Media Player 7.1 .setProxyPort() 1494, 2015
 Windows Media Player 7.1 .setProxySettings() 1494, 2015
 Windows Media Player 7.1 .settings 1479, 2012
 Windows Media Player 7.1 .sourceProtocol 1494, 2012
 Windows Media Player 7.1 .sourceURL 1488, 2012
 Windows Media Player 7.1 .status 1479, 2012
 Windows Media Player 7.1 .stop() 1486, 2015
 Windows Media Player 7.1 .stretchToFit 1479, 2012
 Windows Media Player 7.1 .uiMode 1479, 2012
 Windows Media Player 7.1 .URL 1479, 2012
 Windows Media Player 7.1 .versionInfo 1480, 2012
 Windows Media Player 7.1 .volume 1498, 2012
 Windows Media Player 7.1 .webHelp() 1491, 2015



Windows Media Player 7.1 ActiveX-Control 1470
 Windows Media Player 7.1 aktuelle Playliste 1478, 1490
 Windows Media Player 7.1 aktuelle Playliste hat sich geändert 1480
 Windows Media Player 7.1 aktueller Status 1479
 Windows Media Player 7.1 aktuelles Media Item 1478
 Windows Media Player 7.1 aktuelles media Objekt 1478
 Windows Media Player 7.1 alle Playlisten der Media-Bibliothek 1494
 Windows Media Player 7.1 Anzahl der im System verfügbaren CD-Laufwerke 1484
 Windows Media Player 7.1 Anzahl der Playlisten in der Media-Bibliothek 1496
 Windows Media Player 7.1 Anzahl der Widergaben 1498
 Windows Media Player 7.1 automatische Fehleranzeige 1475, 1497
 Windows Media Player 7.1 autoStart 1475
 Windows Media Player 7.1 Autostart 1479
 Windows Media Player 7.1 Autostart der Widergabe 1497
 Windows Media Player 7.1 balance 1475
 Windows Media Player 7.1 baseURL 1475
 Windows Media Player 7.1 captioningID 1474
 Windows Media Player 7.1 CD-Laufwerke 1483
 Windows Media Player 7.1 CD-Laufwerksbuchstabe 1483
 Windows Media Player 7.1 Collection ID_Player.cdromCollection 1483
 Windows Media Player 7.1 Collection ID_Player.mediaCollection 1491
 Windows Media Player 7.1 Collection ID_Player.playlistCollection 1494
 Windows Media Player 7.1 Control-Elemente 1475, 1478, 1484
 Windows Media Player 7.1 Control-Elemente anzeigen 1479
 Windows Media Player 7.1 Control-Elemente Verfügbarkeit 1486
 Windows Media Player 7.1 currentPosition 1474
 Windows Media Player 7.1 currentMarker 1474
 Windows Media Player 7.1 defaultFrame 1475
 Windows Media Player 7.1 duration 1474
 Windows Media Player 7.1 Eigenschaften vorbelegen in HTML 1474
 Windows Media Player 7.1 Einbettung und Instanzierung im HTML-Dokument 1474
 Windows Media Player 7.1 Einstellungen des Players für die Widergabe 1496
 Windows Media Player 7.1 enableContextMenu 1474
 Windows Media Player 7.1 enabled 1474
 Windows Media Player 7.1 enableErrorDialogs 1475
 Windows Media Player 7.1 endlose Widergabe 1481, 1498
 Windows Media Player 7.1 erzeugen Playliste 1493
 Windows Media Player 7.1 Event buffering 1480
 Windows Media Player 7.1 Event currentItemChange 1480
 Windows Media Player 7.1 Event currentPlaylistChange 1480
 Windows Media Player 7.1 Event error 1480
 Windows Media Player 7.1 Event markerHit 1480
 Windows Media Player 7.1 Event mediaChange 1480
 Windows Media Player 7.1 Event mediaCollectionChange 1481
 Windows Media Player 7.1 Event modeChange 1481

Windows Media Player 7.1 Event openStateChange 1481
 Windows Media Player 7.1 Event playlistChange 1481
 Windows Media Player 7.1 Event playStateChange 1482
 Windows Media Player 7.1 Event positionChange 1482
 Windows Media Player 7.1 Event scriptCommand 1482
 Windows Media Player 7.1 Event statusChange 1483
 Windows Media Player 7.1 Eventbehandlung 1477
 Windows Media Player 7.1 Fehler 1480
 Windows Media Player 7.1 Fehleranzeige 1475, 1497
 Windows Media Player 7.1 Fehlerinformationen 1491
 Windows Media Player 7.1 Feld der Zeiger aller Playlisten 1494
 Windows Media Player 7.1 fullScreen 1474
 Windows Media Player 7.1 für Widergabe bereit 1479
 Windows Media Player 7.1 Hilfe-Seite von Microsoft 1491
 Windows Media Player 7.1 ID_Player Objekt 1478
 Windows Media Player 7.1 ID_Player.cdromCollection Collection 1483
 Windows Media Player 7.1 ID_Player.closedCaption Objekt 1484
 Windows Media Player 7.1 ID_Player.controls Objekt 1484
 Windows Media Player 7.1 ID_Player.currentMedia Objekt 1486
 Windows Media Player 7.1 ID_Player.currentPlaylist Objekt 1488
 Windows Media Player 7.1 ID_Player.error Objekt 1491
 Windows Media Player 7.1 ID_Player.mediaCollection Collection 1491
 Windows Media Player 7.1 ID_Player.network Objekt 1493
 Windows Media Player 7.1 ID_Player.playlistCollection Collection 1494
 Windows Media Player 7.1 ID_Player.settings Objekt 1496
 Windows Media Player 7.1 im HTML-Dokument 1474
 Windows Media Player 7.1 im Internet 1478
 Windows Media Player 7.1 im Netzwerk 1478
 Windows Media Player 7.1 Instanz 1474, 1478
 Windows Media Player 7.1 interner und laufender Status 1479
 Windows Media Player 7.1 Internetverbindung 1493
 Windows Media Player 7.1 invokeURLs 1475
 Windows Media Player 7.1 Kanalausrichtung 1497
 Windows Media Player 7.1 Kontextmenü 1478
 Windows Media Player 7.1 Laden eines Playlisten-Eintrages 1486
 Windows Media Player 7.1 Lautstärke 1498
 Windows Media Player 7.1 Lautstärkeregelung 1479
 Windows Media Player 7.1 Marker 1488
 Windows Media Player 7.1 Media Clip 1469
 Windows Media Player 7.1 Media Objekt 1470
 Windows Media Player 7.1 Media-Bibliothek 1470, 1488
 Windows Media Player 7.1 Media-Bibliothek Anzahl der Playlisten 1496
 Windows Media Player 7.1 Media-Bibliothek Playliste importieren 1496
 Windows Media Player 7.1 Media-Bibliothek und Media-Datei 1493
 Windows Media Player 7.1 Media-Bibliothek verwalten 1491
 Windows Media Player 7.1 Media-Bibliothek alle Playlisten 1494



Windows Media Player 7.1 Media-Datei und Media-Bibliothek 1493
Windows Media Player 7.1 Media-Daten puffern 1480
Windows Media Player 7.1 Medium hat sich geändert 1480
Windows Media Player 7.1 Medium-Datei hat unbekannten Typ 1481
Windows Media Player 7.1 Meta-Datei 1469
Windows Media Player 7.1 Modus der Wiedergabe 1498
Windows Media Player 7.1 Modus der Wiedergabe hat sich geändert 1481
Windows Media Player 7.1 mute 1475
Windows Media Player 7.1 Mute-Taste 1479
Windows Media Player 7.1 name 1474
Windows Media Player 7.1 Name des aktuellen media Objektes 1488
Windows Media Player 7.1 Netzwerkverbindung 1493
Windows Media Player 7.1 Objekt ID_Player 1478
Windows Media Player 7.1 Objekt ID_Player.closedCaption 1484
Windows Media Player 7.1 Objekt ID_Player.controls 1484
Windows Media Player 7.1 Objekt ID_Player.currentMedia 1486
Windows Media Player 7.1 Objekt ID_Player.currentPlaylist 1488
Windows Media Player 7.1 Objekt ID_Player.error 1491
Windows Media Player 7.1 Objekt ID_Player.network 1493
Windows Media Player 7.1 Objekt ID_Player.settings 1496
Windows Media Player 7.1 Pause-Taste 1479
Windows Media Player 7.1 pausieren der Wiedergabe 1486
Windows Media Player 7.1 playCount 1475
Windows Media Player 7.1 Player ist bereit für ist Wiedergabe 1482
Windows Media Player 7.1 Player wartet auf ein Medium 1481
Windows Media Player 7.1 Playereinstellungen für die Wiedergabe 1496
Windows Media Player 7.1 Playerfenster 1479
Windows Media Player 7.1 Playlist von CD 1469
Windows Media Player 7.1 Playliste 1470, 1478, 1488
Windows Media Player 7.1 Playliste aktuelle 1490
Windows Media Player 7.1 Playliste erzeugen 1493
Windows Media Player 7.1 Playliste hat sich geändert 1481
Windows Media Player 7.1 Playliste in die Media-Bibliothek importieren 1496
Windows Media Player 7.1 Playliste ist leer 1480, 1482
Windows Media Player 7.1 Playliste ist offen 1481
Windows Media Player 7.1 Playliste verändern 1490
Windows Media Player 7.1 Playliste wurde erweitert durch anhängen 1480, 1482
Windows Media Player 7.1 Playliste wurde erweitert durch einfügen 1480, 1482
Windows Media Player 7.1 Playliste-Name wurde geändert 1480, 1482
Windows Media Player 7.1 Playlisten-Eintrag laden 1486
Windows Media Player 7.1 Playlisten-Eintrag wechseln 1486
Windows Media Player 7.1 Playlisten-Eintrag wiedergeben 1486

Windows Media Player 7.1 Playlisten-Eintrag wurde gelöscht 1480, 1482
Windows Media Player 7.1 Playlisten-Eintrag wurde verschoben 1480, 1482
Windows Media Player 7.1 Playlisten-Info wurde geändert 1480, 1482
Windows Media Player 7.1 Proxy 1493
Windows Media Player 7.1 puffern von Media-Daten 1480
Windows Media Player 7.1 rate 1475
Windows Media Player 7.1 SAMIFilename 1474
Windows Media Player 7.1 SAMILang 1474
Windows Media Player 7.1 SAMIStyle 1474
Windows Media Player 7.1 schliessen 1480
Windows Media Player 7.1 schnelles Rückspulen während der Wiedergabe 1485
Windows Media Player 7.1 schnelles Vorspulen während der Wiedergabe 1485
Windows Media Player 7.1 Scriptkommandos 1482
Windows Media Player 7.1 Server 1493
Windows Media Player 7.1 shuffle 1481
Windows Media Player 7.1 Skins 1478
Windows Media Player 7.1 Standardbrowser 1480
Windows Media Player 7.1 starten der Wiedergabe 1486
Windows Media Player 7.1 Status der Wiedergabe 1479
Windows Media Player 7.1 Status der Wiedergabe hat sich geändert 1482
Windows Media Player 7.1 Status hat sich geändert 1483
Windows Media Player 7.1 Stereo-Balance 1497
Windows Media Player 7.1 Steuerung der Wiedergabe 1484
Windows Media Player 7.1 stoppen der Wiedergabe 1486
Windows Media Player 7.1 Stop-Taste 1479
Windows Media Player 7.1 stretchToFit 1474
Windows Media Player 7.1 Stummschaltung 1498
Windows Media Player 7.1 uiMode 1474
Windows Media Player 7.1 URL 1474
Windows Media Player 7.1 verändern der Playliste 1490
Windows Media Player 7.1 Verfügbarkeit Control-Elemente 1486
Windows Media Player 7.1 Version 1480
Windows Media Player 7.1 verwalten Media-Bibliothek 1491
Windows Media Player 7.1 Videogröße 1475
Windows Media Player 7.1 Vollbildmodus 1478
Windows Media Player 7.1 volume 1475
Windows Media Player 7.1 Vorbelegung von Eigenschaften in HTML 1474
Windows Media Player 7.1 wechseln des Playlisten-Eintrages 1486
Windows Media Player 7.1 Wiedergabe Anzahl der Wiedergaben 1498
Windows Media Player 7.1 Wiedergabe Autostart 1479, 1497
Windows Media Player 7.1 Wiedergabe benden der Pause 1486
Windows Media Player 7.1 Wiedergabe endlos 1481, 1498
Windows Media Player 7.1 Wiedergabe erfolgt fortlaufend 1479, 1482
Windows Media Player 7.1 Wiedergabe ist gestoppt 1479, 1482
Windows Media Player 7.1 Wiedergabe ist komplett und geendet 1479, 1482



Windows Media Player 7.1 Wiedergabe Kanalausrichtung 1497
 Windows Media Player 7.1 Wiedergabe kann beginnen 1479, 1482
 Windows Media Player 7.1 Wiedergabe Lautstärke 1498
 Windows Media Player 7.1 Wiedergabe mit schnellem Rücklauf 1479, 1482
 Windows Media Player 7.1 Wiedergabe mit schnellem Vorlauf 1479, 1482
 Windows Media Player 7.1 Wiedergabe Modus hat sich geändert 1481
 Windows Media Player 7.1 Wiedergabe Pause benden 1486
 Windows Media Player 7.1 Wiedergabe pausieren 1486
 Windows Media Player 7.1 Wiedergabe pausiert 1479, 1482
 Windows Media Player 7.1 Wiedergabe Playlisten-Eintrag 1486
 Windows Media Player 7.1 Wiedergabe Playlisten-Eintrag wechseln 1486
 Windows Media Player 7.1 Wiedergabe starten 1486
 Windows Media Player 7.1 Wiedergabe Stereo-Balance 1497
 Windows Media Player 7.1 Wiedergabe stoppen 1486
 Windows Media Player 7.1 Wiedergabe Stummschaltung 1498
 Windows Media Player 7.1 Wiedergabe und Playereinstellungen 1496
 Windows Media Player 7.1 Wiedergabe und schnelles Rückspulen 1485
 Windows Media Player 7.1 Wiedergabe und schnelles Vorspulen 1485
 Windows Media Player 7.1 Wiedergabe wechseln des Playlisten-Eintrages 1486
 Windows Media Player 7.1 Wiedergabe zufällig 1481, 1498
 Windows Media Player 7.1 Wiedergabefenster 1479
 Windows Media Player 7.1 Wiedergabemodus 1498
 Windows Media Player 7.1 Wiedergaberate 1498
 Windows Media Player 7.1 Wiedergabestatus 1479
 Windows Media Player 7.1 Wiedergabestatus hat sich geändert 1482
 Windows Media Player 7.1 Wiedergabesteuerung 1484
 Windows Media Player 7.1 WIDTH und HEIGHT 1475
 Windows Media Player 7.1 Windows Media Datei 875
 Windows Media Player 7.1 Windows-Lautstärke-Regelung 1498
 Windows Media Player 7.1 Zeiger 1474, 1478
 Windows Media Player 7.1 zufällige Wiedergabe 1481, 1498
 Windows Media Player in minimaler Version der Control-Elemente 1470, 1474
 Windows Media-Player 909, 914, 922, 927, 945, 971, 1672
 Windows Meta Datei 1469
 Windows Metafile 671, 703
 Windows NT 5.1 412, 1726
 Windows NT 5.x 224
 Windows Script Host 1445
 Windows XP 224, 412, 1726
 Windows XP und JScript 18, 31, 42, 135
 Windows XP und Scriptmaschine 18, 31, 42, 135
 Windows Zwischenablage 277, 378, 476, 1593
 Windows-Betriebssystem 854, 1672
 Windows-Clipboard 1097, 1160
 windowseigene Lautstärkeeinstellung 576

Windows-Lautstärke-Regelung 1498
 Windows-Lautstärke-Regelung Master-Regler 1498
 Windows-Lautstärke-Regelung Media-Regler 1498
 Windowsname logischer 440
 Windows-Taskbar 1182, 1574
 Windows-Taskleiste 438
 Windows-Zwischenablage 1097, 1108, 1160
 Windows-Zwischenablage auslesen 416, 1112, 1781
 Windows-Zwischenablage füllen 416, 1112, 1887
 Windows-Zwischenablage löschen 416, 1112, 1746
 Window-Zwischenablage 1103
 winzig 987, 989
 Wirksamkeit von Attributen 1624
 Wirksamkeit von Attributen im Dokument 426
 with 77
 with Anweisung 99, 150, 209
 with Operator 77
 WMF 671, 703
 Wortanfang mit Grossbuchstabe 815, 825, 1946
 Wortumbruch 609, 1064, 1070, 1662
 Wortumbruch bei Überschreitung der Objektgrenzen 816, 826, 1949
 Wortumbruch Body 590
 Wortumbruch in Textarea 1078, 1733
 w-resize 988
 W-resize 993
 WScript.CreateObject() 1447
 WScript.Shell 1447
 WSH 1445
 Wurzelknoten des Dokumentes 243, 1616
 Wurzelknoten Dokument 425
 wzeichenfolge 1184
 Wzeichenfolge 1184
 X Bitmap 671, 703
 x86 411, 1601
 XBM 671, 703
 xh 872
 X-Koordinate der linken oberen Ecke Body 590, 615, 760, 766
 X-Koordinate der rechten unteren Ecke Body 590, 615, 760, 766
 X-Koordinate der rechten unteren Ecke des Objektes 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1037, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1192, 1664
 X-Koordinate Maus 1107, 1592, 1665, 1684
 x-large 987, 989
 xm 872
 XML 504, 851, 976, 1649, 1723, 1843
 XML Namensraum 866
 xml Objekt 1195
 xml Objekt Events 1142, 1963
 xml Objekt Styles 847
 XML-Dokument 239, 249, 430, 863, 864, 1188, 1195, 1734
 XML-DOM 224, 239, 863, 864, 1188, 1825
 XMLHttpRequest 1369
 XML-Namensraum 479
 XMLNS 246, 247, 479, 482, 505, 506, 507, 562, 563, 568, 572, 573, 590, 598, 602, 615, 619, 628, 649, 660, 663, 667, 682, 687, 688, 692, 696, 697, 700, 704, 709, 713, 714, 718, 722, 726, 727, 731, 736, 740, 744, 745, 750, 751, 754, 760, 761, 766, 767, 775, 776, 1018, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1065, 1071, 1078, 1188, 1192, 1195, 1683, 1709, 1734, 1761



XMLNS-Attribut 239
 XML-Tag im Namensraum 867
 xms 872
 Xray Filter 547
 xs 872
 XSL-Dokument Zeiger 249, 430, 1734
 x-small 987, 989
 x-world/x-vrml 369, 1169, 1170
 xx-large 987, 989
 xx-small 987, 989
 Y-Koordinate der linken oberen Ecke Body 590, 615, 760, 766
 Y-Koordinate der linken oberen Ecke des Objektes 459, 482, 561, 567, 572, 598, 610, 619, 628, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1059, 1064, 1071, 1077, 1083, 1191, 1664
 Y-Koordinate der rechten unteren Ecke Body 590, 615, 760, 766
 Y-Koordinate der rechten unteren Ecke des Objektes 482, 561, 567, 572, 598, 609, 619, 627, 649, 667, 681, 687, 692, 696, 704, 708, 713, 717, 722, 726, 731, 739, 744, 753, 775, 1017, 1028, 1033, 1036, 1042, 1047, 1052, 1058, 1064, 1070, 1077, 1191, 1662
 Y-Koordinate Maus 1107, 1592, 1665, 1684
 zeichen 1183
 Zeichen 57
 Zeichen ' entwerten 56, 58, 114
 Zeichen " entwerten 56, 58, 114
 Zeichen asiatisch Textabstand 815, 824, 1945
 Zeichen nicht druckbar 114
 Zeichen pro Zeile 58
 zeichen* 1183
 zeichen? 1183
 zeichen+ 1183
 zeichenfolge_1|zeichefolge_2 1183
 Zeichenkette 142, 1828, 1911
 Zeichenkette kann über mehrere Zeilen 58
 Zeichenkette mit Ur der nächsten Seite 1165
 Zeichenkette mit Ur der vorhergehenden Seite 1165
 Zeichenkette mit Url der aktuellen Seite 1165
 Zeichenketten 1908
 Zeichenketten verketteten 76
 Zeichenketten verknüpfen 76
 Zeichenkettenkonstante 213
 Zeichenkettensuche 1183, 1186
 Zeichenkettenvariable 213
 Zeichensatz 988
 Zeichensatz des Dokumentes 424, 560, 736, 749, 1187, 1591
 Zeichensatz eines Objektes 424, 560, 736, 749, 1187, 1591
 Zeiger 57, 61, 65, 143, 222, 327, 582, 796, 976, 988
 Zeiger auf Element in einem Formular 597, 614, 633, 634, 635, 636, 637, 638, 641, 686, 691, 696, 700, 703, 708, 713, 717, 721, 726, 730, 753, 759, 765, 1077, 1627
 Zeiger auf aktuelle Objekt-Instanz 77
 Zeiger auf Attribut liefern anhand ID oder NAME 267, 466, 1799
 Zeiger auf das Eltern-Timeline 885, 894, 898, 905, 909, 915, 922, 927, 932, 946, 950, 955, 972, 1713
 Zeiger auf das ERSTE Kind 241, 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 686, 691, 696, 703, 708, 713, 717, 721, 725, 730, 736, 739,

744, 759, 765, 775, 1016, 1028, 1032, 1036, 1041, 1046, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1627
 Zeiger auf das LETZTE Kind 241, 243, 561, 567, 572, 597, 601, 614, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 759, 766, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1649
 Zeiger auf das NACHFOLGENDE Kind 241, 243, 561, 567, 572, 598, 601, 618, 627, 648, 659, 662, 666, 681, 687, 691, 696, 700, 703, 708, 713, 717, 721, 726, 731, 736, 739, 744, 753, 775, 1017, 1028, 1032, 1036, 1041, 1047, 1052, 1058, 1064, 1070, 1077, 1188, 1191, 1660
 Zeiger auf das NACHFOLGENDE Kind Body 590, 614, 760, 766
 Zeiger auf das Vorgängerkind 246, 562, 572, 590, 598, 602, 615, 619, 628, 649, 660, 663, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 760, 766, 1017, 1029, 1033, 1037, 1042, 1048, 1053, 1059, 1077, 1192, 1675
 Zeiger auf das VORHERGEHENDE Kind 241
 Zeiger auf den obersten Knoten des XSL-Dokumentes 249, 430, 1734
 Zeiger auf den Prototyp-Bereich im Objekt 150, 209, 1677
 Zeiger auf Elternelement des Textbereiches 460, 1084, 1850
 Zeiger auf Elternknoten 245, 561, 568, 572, 590, 598, 601, 615, 619, 628, 649, 659, 663, 667, 682, 687, 692, 696, 700, 704, 709, 713, 718, 722, 726, 731, 736, 740, 744, 753, 760, 766, 775, 1017, 1029, 1033, 1037, 1042, 1047, 1053, 1059, 1065, 1071, 1077, 1188, 1192, 1668
 Zeiger auf Filter 509
 Zeiger auf Frame bzw. IFrame 381, 1627
 Zeiger auf Funktionsrumpf 161, 1589
 Zeiger auf HTML-Element 1105
 Zeiger auf HTML-Tag 61
 Zeiger auf Instanz 93
 Zeiger auf Kinder des Objektes 268, 271, 467, 471
 Zeiger auf null 143
 Zeiger auf oberstes Fenster laut Fensterhierarchie 388, 1721
 Zeiger auf TextRectangle-Objekt 460, 483, 564, 569, 574, 591, 600, 602, 611, 616, 620, 629, 683, 689, 693, 698, 705, 710, 715, 719, 723, 728, 732, 737, 741, 746, 755, 762, 768, 777, 1021, 1030, 1034, 1038, 1044, 1049, 1054, 1060, 1067, 1072, 1080, 1084, 1193, 1780
 Zeiger auf übergeordnetes Fenster laut Fensterhierarchie 221, 383, 1667
 Zeiger auf Wurzelknoten (root node) des Dokumentes 243, 1616
 Zeiger auf XML-Dokument 249, 430, 863, 864, 1188, 1195, 1734
 Zeiger Gültigkeit 59
 Zeiger in HTML 61
 Zeiger typisiert 61, 103
 Zeiger und Browserperformance 61
 Zeiger und Datentyp 61
 Zeiger vom Browser erzeugt 59
 Zeiger Zerlegung in Teilzeiger 48
 Zeigerbelegung 70
 Zeigerbildung aus Punktnotation 48, 61
 Zeigererzeugung im Browser 59
 Zeigerfeld dynamisches 43
 Zeigerkopie 60



Zeigertausch 266, 565, 570, 575, 593, 601, 603, 613,
 618, 622, 632, 651, 662, 665, 670, 684, 690, 695, 699,
 702, 707, 712, 716, 721, 725, 729, 734, 738, 742, 748,
 756, 763, 769, 778, 1027, 1032, 1035, 1039, 1045,
 1051, 1056, 1062, 1068, 1074, 1082, 1190, 1195, 1909
 Zeigertyp 58
 Zeigertyp Basis-Datentyp 58
 Zeigertyp Datentyp 58
 Zeigervergleich 143, 149
 Zeigerverwaltung dynamisch 58
 Zeigerverwendung 58
 Zeigerverwendung intern 329
 Zeigerwert 70, 149
 Zeigerzuweisung 59, 69
 Zeile Style 809, 818, 1917
 Zeilenanzahl List-Box 760, 1692
 Zeilenende 53
 Zeilenumbruch 45, 72, 277, 477, 811, 821, 1934
 Zeilenumbruch automatisch 816, 826, 1948
 Zeilenumbruch in Worten 816, 826, 1949
 Zeilenvorschub 56, 58, 113, 114, 447, 1184, 1914
 Zeit und Datum 176
 Zeitfehler 1131, 1999
 Zeitformat UTC 975
 Zeitlinie 865
 Zeitpunkt für Start eines Video 1256
 Zeitpunkte 865
 Zeitsteuerung 865, 887
 Zeitzone 176, 181, 873, 1804
 Zeitzeilen-Berechnungen 176
 Zelle der Tabelle 864
 zentriert 986, 990
 Zertifikate zur Verschlüsselung 284
 ziehen 1103
 Ziel-Fenster 563, 573, 628, 736
 Ziel-Frame 563, 573, 628, 736
 Ziffern-Zeichenketten 1908
 Ziffern-Zeichenkettenwert nach numerisch 200, 1852
 ZigZag Filter 547
 z-index 666, 980, 985
 z-lock 440
 zu anderer HTML-Seite wechseln 1104
 zu anderer Seite wechseln 1104

zu ganzzahlig konvertieren 197, 203
 zu ganzzahlig teilbar konvertieren 198, 203
 ZUERST gefundenes Objekt im Dokument 255, 435,
 827, 1786
 zufälliger Dateiname 1433, 1802
 Zufallszahl 199, 1859
 Zugriff auf Dateisystem per JScript 1430
 Zugriff auf Laufwerk 1434
 Zugriff auf Variable 62
 Zugriff per Alt + Taste 481, 560, 567, 571, 589, 597,
 609, 614, 618, 680, 686, 691, 695, 703, 707, 712, 717,
 721, 725, 730, 743, 752, 759, 774, 1014, 1027, 1032,
 1041, 1046, 1051, 1057, 1063, 1069, 1076, 1190, 1556
 Zugriff per Index 225
 Zugriffe auf das Objekt 225
 Zugriffs auf das Element per Tastaturkürzel 729
 Zugriffsschutz FTP 51
 Zugriffsschutz Password 51
 Zulassen bzw. Sperren von Inhalten auf dem Server für
 Suchmaschinen, die scannen 49
 Zulassen bzw. Sperren von Suchmaschinen 49
 Zurück-Button 863, 1743
 Zustand der Steuertasten 1105
 Zustand des Ladens des Objektes 681, 686, 703, 1595
 Zustand Timeline 866
 zuvorliegende Seite 1165
 Zuweisung 76
 Zuweisungen aller Art 75
 Zuweisungsoperator 76
 Zwangssynchronisierung 911, 917, 924, 930, 934, 948,
 949, 957, 974, 1124, 1131, 1991, 1999
 Zwangs-Synchronisierung von Timelines 879
 Zweck von Cookies 484
 zweidimensionale Matrix 164
 Zwischenablage 277, 413, 476, 1097, 1103, 1108, 1160
 Zwischenablage auslesen 416, 1112, 1781
 Zwischenablage füllen 416, 1112, 1887
 Zwischenablage löschen 416, 1112, 1746
 Zwischenablage von Windows 378, 1593
 Zwischenraum zwischen 2 benachbarten Frames 659,
 1627
 zyklischer Aufruf 401, 1892

